

IUTAM Bookseries

Bernhard Schweizer *Editor*

IUTAM Symposium on Solver-Coupling and Co-Simulation

Proceedings of the IUTAM Symposium on
Solver-Coupling and Co-Simulation, Darmstadt,
Germany, September 18–20, 2017

 Springer

IUTAM Bookseries

Volume 35

The IUTAM Bookseries publishes the refereed proceedings of symposia organised by the International Union of Theoretical and Applied Mechanics (IUTAM).

Every two years the IUTAM General Assembly decides on the list of IUTAM Symposia. The Assembly calls upon the advice of the Symposia panels. Proposals for Symposia are made through the Assembly members, the Adhering Organizations, and the Affiliated Organizations, and are submitted online when a call is launched on the IUTAM website.

The IUTAM Symposia are reserved to invited participants. Those wishing to participate in an IUTAM Symposium are therefore advised to contact the Chairman of the Scientific Committee in due time in advance of the meeting. From 1996 to 2010, Kluwer Academic Publishers, now Springer, was the preferred publisher of the refereed proceedings of the IUTAM Symposia. Proceedings have also been published as special issues of appropriate journals. From 2018, this bookseries is again recommended by IUTAM for publication of Symposia proceedings.

Indexed in Ei Compendex and Scopus.

More information about this series at <http://www.springer.com/series/7695>

Bernhard Schweizer
Editor

IUTAM Symposium on Solver-Coupling and Co-Simulation

Proceedings of the IUTAM Symposium
on Solver-Coupling and Co-Simulation,
Darmstadt, Germany, September 18–20, 2017

Editor

Bernhard Schweizer
Institut für Angewandte Dynamik
Technische Universität Darmstadt
Darmstadt, Germany

ISSN 1875-3507

ISSN 1875-3493 (electronic)

IUTAM Bookseries

ISBN 978-3-030-14882-9

ISBN 978-3-030-14883-6 (eBook)

<https://doi.org/10.1007/978-3-030-14883-6>

Library of Congress Control Number: 2019935837

© Springer Nature Switzerland AG 2019

This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.

The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

The publisher, the authors and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, expressed or implied, with respect to the material contained herein or for any errors or omissions that may have been made. The publisher remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

This Springer imprint is published by the registered company Springer Nature Switzerland AG.
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

Organizing Committee

Chair

Bernhard Schweizer
Institute of Applied Dynamics
TU Darmstadt
Darmstadt, Germany

Members

Jan Kraft
Tobias Meyer
Maria Rauck
Helga Lorenz
Daixing Lu

All from:
Institute of Applied Dynamics
TU Darmstadt
Darmstadt, Germany

Scientific Committee

Chair

Prof. Bernhard Schweizer
Institute of Applied Dynamics
TU Darmstadt
Darmstadt, Germany

Members

Prof. Jorge Ambrosio
IDMEC/IST
Instituto Superior Tecnico
Lisboa, Portugal

Prof. Javier Cuadrado
Laboratory of Mechanical Engineering
University of La Coruña
Ferrol, Spain

Prof. Aki Mikkola
Machine Design Mechanical Engineering, LUT School of Energy Systems
Lappeenranta University of Technology
Lappeenranta, Finland

Prof. Dan Negrut
Mechanical Engineering
University of Wisconsin-Madison
Madison, USA

Prof. Bernhard Schweizer
Institute of Applied Dynamics
TU Darmstadt
Darmstadt, Germany

Prof. Taichi Shiiba
Vehicle Dynamics laboratory
Meiji University
Tokyo, Japan

Prof. Bernd Simeon
Department of Mathematics
TU Kaiserslautern
Kaiserslautern, Germany

IUTAM Representative on Scientific Committee

Prof. Peter Eberhard
University of Stuttgart
Germany

Preface

Co-simulation—also called solver or simulator coupling—is a frequently used numerical technique to couple two or more solvers in time domain. One major field of application for co-simulation methods is the analysis of multi-disciplinary problems.

Usually, specialized simulation codes exist for different physical disciplines, e.g., FEM codes for structural dynamics analyses, CFD codes for fluid dynamic problems, or multibody codes for the dynamic analysis of mechanisms. In order to simulate a coupled multidisciplinary problem, the different codes can be coupled by means of an appropriate co-simulation approach. Simulator coupling is, for instance, successfully applied in the field of fluid/structure interaction, for coupling multibody and hydraulic systems or in the analysis of electromechanical systems. Solver coupling may, however, also be used to analyze monodisciplinary problems in order to parallelize the simulation process.

On the one hand, the symposium focused on recent advances in the development of numerical methods for solver coupling. Of current interest are—among others—the following subjects:

- New explicit, implicit, and semi-implicit co-simulation methods (with improved efficiency, accuracy, and stability behavior).
- New approaches for realizing variable communication-time grids.
- Advances in the stability and convergence analysis of solver coupling methods.

On the other hand, the symposium intended to pick up recent developments in the practical application of co-simulation methods. Of present interest are, for instance, the following topics:

- New fields of application for solver coupling approaches.
- New developments in the parallelization of dynamic models with co-simulation techniques.
- Standardization of co-simulation interfaces, i.e., standardization of data and model exchange.

Bringing together experts in these different fields from many working groups from all over the world enabled us to review the state-of-the-art, to discuss further activities, to open problems, and to promote common research initiatives for the future.

Darmstadt, Germany
2017

Bernhard Schweizer

Contents

1	Relaxing Stiff System Integration by Smoothing Techniques for Non-iterative Co-simulation	1
	Martin Benedikt and Edo Drenth	
2	TLM-Based Asynchronous Co-simulation with the Functional Mockup Interface	27
	Robert Braun, Robert Hällqvist and Dag Fritzon	
3	Local Extrapolation and Linear-Implicit Stabilization in a Parallel Coupling Scheme	43
	Michael Burger and Stefan Steidel	
4	Performance Improvement of Explicit Co-simulation Methods Through Continuous Extrapolation	57
	Martin Busch	
5	Stable Adaptive Co-simulation: A Switched Systems Approach	81
	Cláudio Gomes, Benoît Legat, Raphaël M. Jungers and Hans Vangheluwe	
6	The SNI-MoWrapper: An FMI-Compatible Testbed for Numerical Algorithms in Co-simulation	99
	Stefan Hante, Martin Arnold and Markus Köbis	
7	A Coupled Finite Element Analysis Approach Combining In-House and General-Purpose Codes	117
	Tomoyoshi Horie, Tomoya Niho and Daisuke Ishihara	
8	Reduction of the Computation Time of Large Multibody Systems with Co-simulation Methods	131
	Jan Kraft, Tobias Meyer and Bernhard Schweizer	

- 9 Explicit Co-simulation Approach with Improved Numerical Stability 153**
Pu Li, Daixing Lu, Robert Schmoll and Bernhard Schweizer
- 10 The Influence of Secondary Flow on the Dynamics of Vibrating Tubes 203**
J. P. Meijaard
- 11 Error Estimation Approach for Controlling the Communication Step-Size for Explicit Co-simulation Methods 217**
Tobias Meyer, Jan Kraft, Daixing Lu and Bernhard Schweizer
- 12 Stability and Error Analysis of Applied-Force Co-simulation Methods Using Mixed One-Step Integration Schemes 243**
Bryan Olivier, Olivier Verlinden and Georges Kouroussis
- 13 A Strategy to Conduct Numerical Simulation of Wind Turbine Considering the Soil-Structure-Interaction by Using a Coupled FEM-SBFEM Approach in Time Domain 255**
Marco Schauer, Francesca Taddei and Sissy Morawietz
- 14 Constraint Coupling for Flexible Multibody Systems: Stabilization by Modified Spatial Discretization 269**
Fabio Schneider and Michael Burger
- Author Index 291**

Chapter 1

Relaxing Stiff System Integration by Smoothing Techniques for Non-iterative Co-simulation



Martin Benedikt and Edo Drenth

Abstract Non-iterative or weak-coupling is the most applicable scheme for the co-simulation of interacting subsystems, where subsystems are solved independently with data exchange at restricted time instants. This contribution analyzes the continuous co-simulation from a different, a system-oriented, point of view and three coupling challenges are identified: co-simulation discretization error, sampling and discontinuities introduced. Introduction of smoothing filters can be interpreted as an additional co-simulation discretization error and affects the entire system behavior in general. However, energy-preservation-based considerations has proven to improve co-simulation performance, enabling filter applications according to the communication step-size, where mitigated frequency parts are added by the recently proposed correction schemes. This way, *numerical stiffness* is relaxed by an energy preserving mapping of high frequencies into low frequency ranges, based on *Parseval's* identity. The proposed approaches are demonstrated along a theoretical as well as an industrial co-simulation example.

1.1 Introduction to Co-simulation

Satisfying increasing customer needs leads to more complex (large) systems and as a consequence a comprehensive system design is required. Nowadays, the system design process and the analysis itself are typically carried out in a computer-aided

M. Benedikt (✉)

Virtual Vehicle Research Center, AreaE E/E and Software, Inffeldgasse 21A,
8130 Graz, Austria
e-mail: martin.benedikt@v2c2.at

E. Drenth

Volvo Car Corporation, Requirements & Operational Development, Torslanda PVP2,
40531 Goteborg, Sweden
e-mail: edo.drenth@volvocars.com

© Springer Nature Switzerland AG 2019

B. Schweizer (ed.), *IUTAM Symposium on Solver-Coupling and Co-Simulation*,
IUTAM Bookseries 35, https://doi.org/10.1007/978-3-030-14883-6_1

manner using specialized simulation tools. But the classical approach of modeling and solving the system under investigation in a single simulation tool may not be possible for any system design because of special limitations of the used simulation tool. For example, the used modeling language may restrict a detailed system description or problem tailored solvers are not available. As a consequence, teams of engineers use domain-specific simulation tools which are tailored to a dedicated technical domain (e.g. mechanical, electrical, thermal, uid, ...) and develop innovations according to a subproblem of the overall complex system which is herein termed as subsystem. The main drawback of this methodology is the missing (virtual) interaction of the subsystems during the virtual system design and analysis process. Establishing connections between the heterogeneous subsystems is strongly recommended in order to achieve reliable estimates for the entire dynamic system behavior. Thus, the development and analysis of large systems leads to coupled modular simulations referred to as co-simulation. It's modular approach enables the seamless exchange and reuse of models ensuring return of invest.

By this co-simulation approach, each subsystem is modeled and solved independently within a specific domain using a customized simulation tool. Most often the assembly of two simulators out of different engineering domains is sufficient for analysis of specific problems but for large and complex systems, which frequently occur in the automotive industry, e.g. vehicle simulation for stabilizing controller design, more than two simulation tools have to be integrated. For both scenarios so called co-simulation platforms provide interfacing capabilities for relevant simulation tools and take care about the handling of coupling data at pre-defined time instants for synchronization purposes, see [1–3]. In addition, with the increasing interest in co-simulation by the industry, a specification for standardized exchange of subsystem simulation models was developed. This standard is called *Functional Mockup Interface* (FMI) and defines a dedicated interface for *Functional Mockup Units* (FMUs) of simulation models exclusively or simulation subsystems including the model and the tailored numerical solver, whose interfaces are referred to as *FMI for Model-Exchange* or *FMI for Co-Simulation*, respectively. Currently, FMI 2.0 specification is available [4, 5].

According to the continuous time co-simulation approach in general, there exist two major coupling methodologies: non-iterative and iterative coupling schemes [6, 7]. Within both schemes the involved subsystems are solved independently over defined time intervals, the so-called macro-time steps, using their implemented numerical solvers with fixed or variable step-sizes, i.e. the micro-step-sizes. After each macro-time step synchronization of the subsystems is carried out via the exchange of coupling data, specially the inputs and the outputs of the interconnected subsystems [8–10]. In contrast to the non-iterative scheme, iterative schemes solves the subsystems for several times over the same macro-time step until a defined criterion for termination is reached, e.g. the number of performed iterations or w.r.t. a metric for assessing simulation accuracy. In case of convergence to the unique solution, the iterative scheme ensures highly accurate co-simulation results [13]. However, there exist one major disadvantage: resetting the involved simulation tools for each iteration step is necessary. Thus, besides iterative once, some examples

are reported in literature [11], non-iterative schemes are of high interest in practical applications, as resetting of models and simulators are barely supported by simulation tools today. On the other hand, by utilizing non-iterative schemes, extrapolation of coupling quantities is mandatory for solving bidirectional dependencies between subsystems, i.e. solving the causality problem.

Furthermore, co-simulation is in general motivated by enabling the integration of rather complex simulation subsystems, such as full vehicle dynamics or drive-line simulations, including significant stiffness. The classical use of (the explicit) non-iterative schemes require very small coupling step-sizes or simply fail. An applicable solution for stiff systems is represented by linearly-implicit approaches (Rosenbrock-type methods [12]), which were already successfully applied to non-iterative real-time simulation of stiff ordinary differential equation (ODE) and differential algebraic equation (DAE) systems in the past [13, 14]. This approach and others reported in [15–19] are based on additional information about dedicated partial derivatives of the individual subsystems, which is specified by FMI 2.0, but basically barely supported by simulation tools.

The Transmission Line Method (TLM) [20] or the promising Quantized State Simulation (QSS) approach [21] are currently in discussion within the scientific community and herein mentioned for completeness. The TLM motivates introduced time delays by physical effects and the QSS approach renders the continuous time co-simulation to a discrete event simulation approach, which is based on a new kind of numerical solvers to be implemented within the individual subsystems. Both approaches goes inline with a modification of the subsystems.

This contribution investigates continuous time co-simulation from a different, a system-oriented, point of view. The remainder of the article is organized as follows: the next section introduces the challenges related to non-iterative co-simulation. On this basis, in Sect. 1.3, a recently proposed energy-preserving strategy is revised and used for implementation of smoothing filters. With a special focus on sampling effects derived energy-preserving smoothing filters for handling aliasing effects in cases of stiff and extra-stiff subsystem co-simulation are discussed. Section 1.4 demonstrates the enhancements on a theoretical example and within Sect. 1.5 the proposed methods are applied to an relevant industrial co-simulation example.

1.2 Challenges in Non-iterative Co-simulation

For sake of simplicity in implementation and due to restricted simulation tool interfaces, non-iterative schemes are state of the art in co-simulation applications today. As mentioned above, each subsystem is independently solved by a tailored fixed- or variable step-size solver, whereas herein the local steps are referred to as micro-time steps $\delta T^{<l>}$, the superscript denotes the l th time-step [22]. For synchronization purposes coupling data is exchanged at specific coupling points in time $t_{\Delta}^{<M>}$:

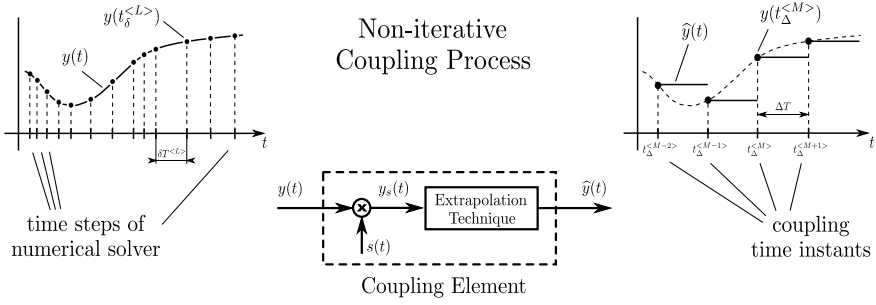


Fig. 1.1 Coupling process comprising sampling and extrapolation

$$t_{\Delta}^{<M>} = t_{start} + \sum_{m=1}^M \Delta T^{<m>}, \quad (1.1)$$

where $\Delta T^{<m>}$ represents the m th macro-time step; with a typical offset $t_{start} = 0$. In the most general case, instead of updating exclusively the value at the coupling time instant, i.e. the individual macro-time step ends, all intermediate output values determined by the numerical solver of the subsystem in the time interval of the last macro-time step are exchanged. This additional information enables advanced co-simulation algorithms an improved performance. For example, the values at micro-steps could lead to improved extrapolation capabilities [23]. Figure 1.1 illustrates the relation of micro- and macro-time steps as well as indicates the coupling process [24]. The subsystem model is solved independently by the corresponding numerical solver and the resulting output signal $y(t)$ is used by the co-simulation algorithm to determine inputs $\hat{y}(t)$ for any of the interconnected subsystems by extrapolation. Extrapolation is mandatory due to the modular character of the co-simulation and the exclusive data communication at coupling time instances for solving the resulting causality problem. This co-simulation approach is referred to as *weak coupled* co-simulation—the following challenges arises:

- **Co-Simulation Discretization Error:** for solving the causality problem extrapolation of output signals is applied to predict the future progress over the interval of the macro-time step. Industrial practice is to keep the last known value of the coupling signal constant for the actual macro-time step. This extrapolation is known as zero-order-hold (ZOH) extrapolation and is pointed out in Fig. 1.1 (right). As illustrated, due to the mandatory and piecewise extrapolation of the coupling signal a Co-Simulation Discretization Error—representing an estimation error—is introduced which affects the overall system behavior.
- **Aliasing Effects:** in order to cope with subsystem-dependent macro-step sizes the coupling data, i.e. the discrete values at macro- and/or micro-time steps (intermediate output values), is interpreted as a continuous time coupling signal. Thus, sampling at coupling time instants of the continuous time coupling signal is mandatory and *aliasing effects* may occur [23]. With respect to implementation details,

required coupling signal values at coupling time instants are determined based on the linear or higher-order interpolation schemes.

- **Discontinuities at coupling time instances:** the concept of piecewise extrapolation of coupling signals introduces discontinuities at coupling time points, see Fig. 1.1 (right). Discontinuities introduce high frequency parts and coupling signals possess enlarged bandwidths [23, 24]. In consequence, numerical accuracy of applied fixed step-size solvers is degraded or the number of performed micro-steps of variable step-size solvers is increased [25]. More critically, discontinuities with its high frequency components excite existing fast dynamics of subsystems.

After this rough introduction into non-iterative co-simulation the three arising challenges are detailed within the subsequent sections.

Remark During co-simulation of the individual subsystems numerical values, approximating the exact solution, are determined at discrete points in time ($t_{\delta}^{<L>}$) by the numerical solver. These values are exchanged between subsystems during co-simulation at communication point instances and therefore a multi-rate analysis of the overall co-simulation scheme would be a possible choice. But complex co-simulation scenarios requires fixed and variable step-sizes on micro- and macro-time step level, which would lead to a very extensive analysis of an discrete time multi-rate sampled system. However, by interpretation of the coupling signals as a time continuous signal the analysis is significantly simplified, as the analysis can be conducted within the continuous time and frequency domain. With respect to this, the following sections analyze non-iterative co-simulation within the continuous time and frequency domain and, for sake of simplicity, for the consecutive part of this article exclusively a constant macro-step-size $\Delta T = \Delta T^{<m>} \forall m$ is considered for analysis.

1.2.1 Sampling of the Coupling Signal

Subsystems are solved by tailored numerical solvers and thereby the exact solution of the individual subsystems are approximated at dedicated points in time $t_{\delta}^{<l>}$ based on micro-time steps. However, by interpretation of a subsystem's output as continuous time signal, i.e. artificially ideal signal reconstruction, sampling of the signal is necessary for application of extrapolation techniques. The process of sampling can be described by modulation of the signal by an impulse train:

$$s(t) = \sum_{n=-\infty}^{\infty} \delta(t - n\Delta T), \quad (1.2)$$

where $\delta(t)$ denotes a continuous time impulse function leading to the mathematical representation of the resulting sampled coupling signal

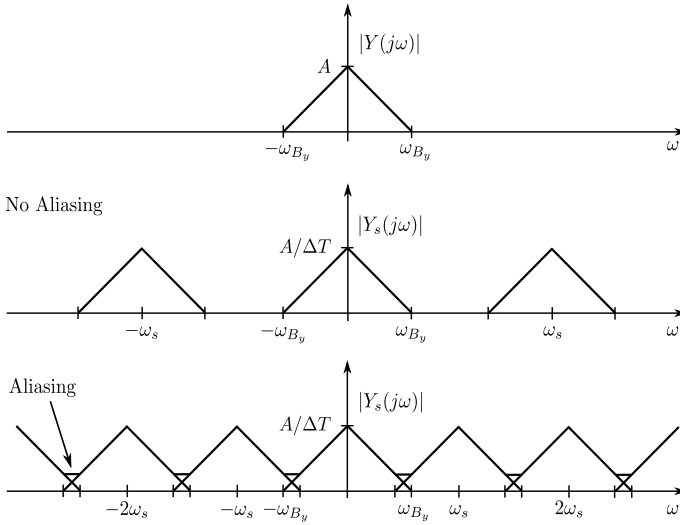


Fig. 1.2 Effect of sampling in frequency domain

$$y_s(t) = y(t)s(t) = y(t) \sum_{n=-\infty}^{\infty} \delta(t - n\Delta T), \quad (1.3)$$

which is subsequently used for extrapolation, see Fig. 1.1. For modeling purposes the impulse sampled signal (1.3) is described within the frequency domain utilizing the *Fourier* transformation [26]:

$$Y_s(j\omega) = \frac{1}{\Delta T} \sum_{n=-\infty}^{\infty} Y(j\omega - nj\omega_s) \quad (1.4)$$

Obviously, the spectrum $Y_s(j\omega)$ of the impulse sampled signal $y_s(t)$ is represented by the $1/\Delta T$ scaled periodically repeated spectrum $Y(j\omega)$ of the unsampled signal $y(t)$; repeated by $\omega_s = 2\pi f_s$ with the sampling frequency $f_s = 1/\Delta T$. Figure 1.2 depicts this effect, where a bandwidth-limited input signal $y(t)$, with $|Y(j\omega)| = 0$ for $\omega \geq |\omega_{B_y}|$, is sampled by ΔT resulting in a periodical spectrum scaled by $1/\Delta T$. The middle illustration in Fig. 1.2 shows the case where no aliasing happens. In the bottom one, the scaled spectra overlaps and aliasing occurs.

Notably, large macro-step-sizes lead to overlapping spectra of the input signals with the consequence that higher frequency components are mirrored into lower frequency regions. This effect is called *aliasing* whereas the original coupling signal will be distorted. In fact, due to sampling *aliasing* can occur which influences the dynamic behavior and consequently defines limits for the macro-step-size. Furthermore, the *Fourier* transform of the impulse sampled signal (1.4) consists of an

infinite sum and is therefore difficult to analyze. In order to simplify the analysis the following assumption is posed:

Assumption 1.2.1 The macro-step-size is selected to avoid *aliasing* w.r.t. a limited bandwidth output signal and exclusively the original bandwidth is relevant.

Following Assumption 1.2.1 the spectrum of the impulse sampled signal is truncated and only considered in the bandwidth of the output coupling signal:

$$Y_s^*(j\omega) = \frac{1}{\Delta T} Y(j\omega). \quad (1.5)$$

In addition, assuming that the extrapolation process can be modeled by a transfer function $G(s)$, the overall coupling element is described by the following relation:

$$\widehat{y}(s) = H(s)y(s) \quad \text{with} \quad H(s) = \frac{1}{\Delta T} G(s), \quad (1.6)$$

where s denotes the *Laplace* variable, with $y(s) = Y(j\omega)|_{j\omega=s}$.

1.2.2 Co-simulation Discretization Error

Most often the zero-order extrapolation scheme is used for extrapolation: unknown inputs are treated as constant quantities over the interval of the actual constant macro-time step ΔT equal to the value of the coupling signal $y(t_{\Delta}^{<M>})$ at the actual coupling time instant $t_{\Delta}^{<M>}$:

$$\widehat{y}(t) = y(t_{\Delta}^{<M>}) \quad \text{with} \quad t_{\Delta}^{<M>} \leq t < t_{\Delta}^{<M+1>}. \quad (1.7)$$

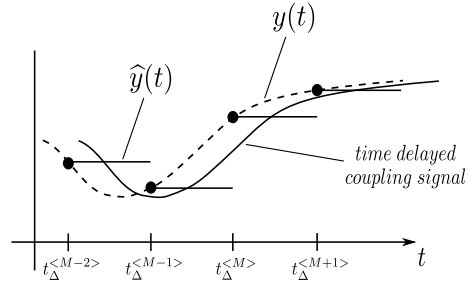
This extrapolation scheme results in a piecewise continuous time signal $\widehat{y}(t)$ describing a piece-wise constant function. Utilizing the *Laplace* transformation and appropriate scaling regarding required sampling of the coupling signal leads to a transfer function $H(s)$

$$H(s) = \frac{\widehat{y}(s)}{y(s)} = \frac{1 - e^{-s\Delta T}}{s\Delta T}, \quad (1.8)$$

which describes the behavior of the coupling element in the special case of zero-order extrapolation [24]. After substitution of the *Laplace* variable $s = j\omega$ and simple trigonometric substitutions the frequency response of the coupling element may be written as follows:

$$H(j\omega) = \frac{2 \sin(\omega\Delta T/2)}{\omega\Delta T} e^{-j\omega\frac{\Delta T}{2}} = \underbrace{\overline{H}(\omega)}_{\text{real}} e^{-j\omega\frac{\Delta T}{2}} \quad (1.9)$$

Fig. 1.3 Zero-order extrapolation at coupling time instants and resulting time delayed coupling signal [24]



As expected and derived by Eq. (1.9), due to the zero-order extrapolation scheme a time-delay of half the macro-step-size is introduced. In Fig. 1.3, the effect of the simple coupling scheme is illustrated. Given a continuous time coupling signal $y(t)$ a piece-wise constant function $\hat{y}(t)$ is generated by the coupling element.

This sample and hold extrapolation scheme limits the macro-step-size significantly when subsystems with stiff dynamics have to be considered and high accuracy is required. Especially in closed loop configurations, e. g. control systems, the artificially introduced time-delay influences significantly the dynamics of the coupled system and may lead to an unstable system behavior. As a consequence, small step-sizes have to be chosen resulting in a rapidly increasing overall simulation time.

1.2.3 Discontinuities at Coupling Time Instances

Discontinuities at coupling time instants occur due to extrapolation and represent high frequency components of the coupling signal. On the one hand, these discontinuities may influence the accuracy and the behavior of the implemented numerical solvers and, on the other hand, may excite existing fast dynamics of the subsequent subsystem [23, 27]. Regarding numerical accuracy consider the following example of a time-dependent dynamical subsystem:

$$\frac{dy_i(t)}{dt} = f_i(y_i(t), u_i(t), t), \quad (1.10)$$

where $y_i(t)$ and $u_i(t)$ denotes the input and output of the i th subsystem, respectively. During non-iterative co-simulation the input of the subsystem is estimated using the output of the previous j th subsystem $u_i(t) = \hat{y}_j(t)$ which is disturbed in general. The influence on the quality of the numerical solution can be shown using the *Taylor* series expansion in the vicinity of t_0 . For the subsystem (1.10) above it follows:

$$y_i(t_0 + h) = y_i(t_0) + \left. \frac{dy_i(t)}{dt} \right|_{t=t_0} h + \left. \frac{d^2 y_i(t)}{dt^2} \right|_{t=t_0} \frac{h^2}{2!} + \dots \quad (1.11)$$

Using the explicit *Euler* scheme the *Taylor* expansion is truncated after the linear term. A significant error is therefore composed by the first neglected term of the series:

$$\left. \frac{d^2 y_i}{dt^2} \right|_{t=t_0} \frac{h^2}{2!} = \frac{\partial f_i}{\partial y_i} \frac{dy_i}{dt} + \frac{\partial f_i}{\partial u_i} \frac{du_i}{dt} + \left. \frac{\partial f_i}{\partial t} \right|_{t=t_0} \frac{h^2}{2!}. \quad (1.12)$$

Obviously, in this representation the derivative of the system input $u_i(t)$, i.e. the extrapolated output signal $\widehat{y}_j(t)$ of the previous subsystem, appears. In the case of the explicit *Euler* scheme discontinuities at coupling time instants significantly influence the accuracy of the numerical solution. For numerical algorithms using variable step-sizes discontinuities may affect the step-size, i.e. the micro-step-size, leading to an enhanced number of required steps and larger simulation times. This means that the quality of the non-iterative coupling process also depends on the characteristics (dynamics and solver) of the subsequent subsystem.

1.3 Addressing Co-simulation Challenges

For addressing the identified non-iterative co-simulation challenges a concept based on energy-preservation considerations was recently developed [28, 29]. This approach is motivated by the mitigation of dissipating and/or producing energy, due to the co-simulation discretization error, within the couplings between the co-simulated subsystems. As the co-simulation discretization error can be determined exclusively after the macro-time step at the coupling time instant, modification of the extrapolation over the subsequent macro-time step can be applied for compensation of energy transmission errors. From an system oriented point of view this concept is equivalent to fulfilling the constraint:

$$H(j\omega) = 1 + j0, \quad \forall \omega, \quad (1.13)$$

which is impossible in general. However, the developed strategy modifies extrapolations to ensure at least a limited bandwidth fulfilling this constraint. This bandwidth is herein referred to as the *efficient bandwidth* of the coupling element. As simulation represents an approximation in general and Eq. 1.13 is approximated in a specific, lower frequency range the proposed concept is called *nearly energy-preserving coupling element* (NEPCE). Figure 1.4 depicts its overall structure.

1.3.1 Handling Co-simulation Discretization Errors

Extrapolation is performed for solving bidirectional dependencies between subsystems. As extrapolation is directly associated to estimation an estimation error, i.e. the co-simulation discretization error, is introduced, which affects the entire dynamic

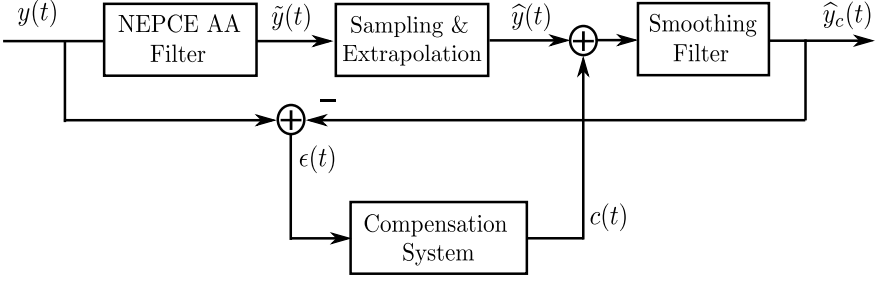


Fig. 1.4 Overall structure of the energy-preserving coupling element

behavior of the overall system. In non-iterative co-simulation the deviation $\epsilon(t)$ between the extrapolation $\hat{y}(t)$ and the co-simulation result $y(t)$:

$$\epsilon(t) := y(t) - \hat{y}(t), \quad (1.14)$$

can be determined at the coupling time instances over the the interval of the last macro-time step. As no iterations are performed this information can exclusively¹ be used to modify the result at the coupling time instant or the extrapolation $\hat{y}(t)$ over the subsequent macro-time step by a correction signal $c(t)$, leading to a modified extrapolation $\hat{y}_c(t)$:

$$\hat{y}_c(t) = \hat{y}(t) + c(t). \quad (1.15)$$

The introduced correction signal is determined based on energy-preservation considerations. In a simple approach the co-simulation discretization error is integrated over the last macro-time step and used for construction of the correction signal for the next macro-time step. The following equation represents an example for realization of a *constant correction* signal [24]:

$$c(t) := \frac{1}{\Delta T^{<m+1>}} \int_{t_{\Delta}^{>m-1>}}^{t_{\Delta}^{<M>}} \epsilon(\tau) d\tau \quad \text{with} \quad t \in (t_{\Delta}^{<M>}, t_{\Delta}^{<M+1>}]. \quad (1.16)$$

This specific realization of the correction is based on the assumption that the co-simulation discretization error equals over subsequent macro-time steps, which is violated in general for co-simulation of dynamical subsystems. For handling this issue adjusting the macro-step-size and tuning of the proposed compensation system [28] or the use of additional subsystem information, e.g. by utilizing dedicated time derivatives of output signals or partial derivatives [18, 19], is proposed. However, in the herein described specific case (1.16) a *constant correction* is realized. The realization of the correction signal is based on *Parseval's identity*, which states an important link between the time domain and the frequency domain [30]:

¹Herein it is assumed that no further co-simulation capabilities are supported by the simulation tools, as for example the provision and utilization of partial derivatives as specified in FMI 2.0.

$$\int_{-\infty}^{\infty} |x(t)|^2 dt = \frac{1}{2\pi} \int_{-\infty}^{\infty} |X(j\omega)|^2 d\omega, \quad (1.17)$$

where $x(t)$ and $X(j\omega)$ denote an arbitrary signal in time domain and its spectrum in frequency domain, respectively. In other words, the correction signal can be realized individually, ensuring the same (!) frequency content as well as the same (!) energy in terms of *signal energy*.

Note: In case of applying smoothing filters, these filters are considered by the synthesis of the compensation system for enhancing compensation performance. For sake of simplicity, the compensation system discussed above is designed for application without a smoothing filter. For further information see [29].

1.3.2 Handling Aliasing Effects

Aliasing effects are caused by sampling of a continuous time signal, as discussed and illustrated in Sect. 1.2.1. The classical approach to cope with this issue is the application of low-pass filters to restrict the bandwidth of the related signal to a certain range prior to sampling [26, 30]. These classical anti-aliasing filters introduce, among other things, an additional phase-shift, which disturbs the entire behavior of the overall dynamical system. In the proposed concept (Fig. 1.4) the anti-aliasing filter is implemented before the sampling and extrapolation stage and thus, the effect of the anti-aliasing filter itself is considered by the correction scheme, meaning the anti-aliasing filter is interpreted as an additional co-simulation discretization error.

Relaxing numerical stiffness—As the phase-shift of anti-aliasing filters relates to the order of the filter, typically filters of low order are applied. As a consequence, not all frequency components are completely mitigated and, depending on the chosen macro-step-size, some frequency components are mirrored into low frequency ranges causing aliasing. This effect limits the macro-step-size and leads to a decreased co-simulation performance. In order to circumvent this problem the above outlined idea for realization of the correction signal (1.16) can be utilized. In contrast to the compensation of the co-simulation discretization error, where the correction has to be applied at the coupling time instant or the subsequent macro-time step, in case of anti-aliasing, the output signal can be reshaped over the whole past macro-time step w.r.t. the *Parseval's identity* (1.17). A dedicated realization results in an equation of the form:

$$c(t) = c(t_{\Delta}^{<M-1>}) + \frac{1}{(\Delta T^{<m>})^2} \left(\int_{t_{\Delta}^{<M-1>}}^{t_{\Delta}^{<M>}} \epsilon(\tau) - c(t_{\Delta}^{<M-1>}) d\tau \right) t \quad (1.18)$$

$$\text{with } t \in (t_{\Delta}^{<M-1>}, t_{\Delta}^{<M>}],$$

representing an energy-preserving anti-aliasing filter, the NEPCE Anti-Aliasing Filter, where the realization of the *linear correction* signal is extended by an internal state storing the last correction value. Setting $\widehat{y}(t) = 0, \forall t$ in Eq. 1.14 and replacing $c(t) = \widetilde{y}(t)$ leads to the final energy-preserving anti-aliasing filter equation:

$$\widetilde{y}(t) = \widetilde{y}(t_{\Delta}^{<M-1>}) + \frac{1}{(\Delta T^{<m>})^2} \left(\int_{t_{\Delta}^{<M-1>}}^{t_{\Delta}^{<M>}} y(\tau) - \widetilde{y}(t_{\Delta}^{<M-1>}) d\tau \right) t \quad (1.19)$$

with $t \in (t_{\Delta}^{<M-1>}, t_{\Delta}^{<M>}]$.

As sampling is involved instead of a transfer function the resulting spectrum of the output signal $\widetilde{y}(t)$ including aliasing components renders to:

$$\widetilde{Y}(j\omega) = \underbrace{\frac{e^{j\omega\Delta T} + e^{-j\omega\Delta T} - 2}{(j\omega)^2 \Delta T}}_{\text{linear interpolation}} \frac{1 - e^{-j\omega\Delta T}}{\Delta T \xi} \sum_{n=-\infty}^{\infty} \frac{\xi}{\Delta T} \frac{1}{j\omega} Y(j\omega - nj\omega_s), \quad (1.20)$$

where the integrated input signal is sampled by a fraction of the macro-step-size, realized by the factor $\xi \in \mathbb{N}$, leading to the sampling frequency $\omega_s = 2\pi\xi/\Delta T$.

A very similar approach were developed and published recently, proposing the use of a continuous time moving-average (CMA) filter [31]. The idea is to implement the filter:

$$\widehat{y}(t) = \frac{\bar{y}(t_{\Delta}^{<M>}) - \bar{y}(t_{\Delta}^{<M-\vartheta>})}{\vartheta}, \quad \bar{y}(t) = \frac{1}{\Delta T} \int y(t) dt, \quad (1.21)$$

with $t \in (t_{\Delta}^{<M>}, t_{\Delta}^{<M+1>}]$,

as an energy-preserving anti-aliasing filter at the individual continuous time outputs within the subsystems, with $\vartheta \in \{1, 2\}$. The average filter of second order ($\vartheta = 2$) is used for non-iterative co-simulation; first order ($\vartheta = 1$) is devoted for iterative co-simulation. As the filter outputs are defined at discrete points in time exclusively, the output values are kept constant for the next macro-time step which allows for an interpretation as ordinary zero-order-hold (ZOH) extrapolated inputs to connected subsystems. The resulting spectrum after the connector element, including the continuous time moving-average filter and ZOH extrapolation can be determined by the following equation ($\omega_s = 2\pi/\Delta T$):

$$\widehat{Y}_{CMA}(j\omega) = \underbrace{\frac{1 - e^{-j\omega\Delta T}}{j\omega}}_{\text{ZOH}} \frac{1 - e^{-j\omega\vartheta\Delta T}}{\vartheta\Delta T} \sum_{n=-\infty}^{\infty} \frac{1}{\Delta T} \frac{1}{j\omega} Y(j\omega - nj\omega_s). \quad (1.22)$$

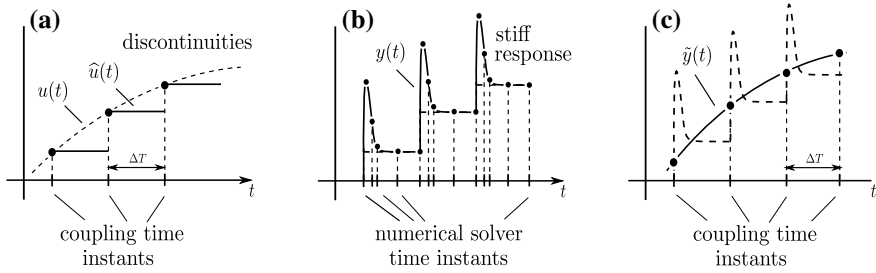


Fig. 1.5 Excitation of high dynamics by the (zero order hold) extrapolated input signal $\hat{u}(t)$, the stiff subsystem response $y(t)$ and the corrected output signal $\tilde{y}(t)$

Instead of ZOH extrapolation, this approach proposes the implementation of a signal-reconstruction filter, representing physics-based low-pass filters,² at the inputs within the subsequent subsystem to reconstruct the continuous time signal from the discrete time series, i.e. the samples at the coupling time instants. The combination of the continuous time moving-average filter (CMA) and the signal reconstruction filter (EPF) is very similar to Eq. 1.20 and constitutes the so-called energy-preserving connector element (EPCE).

By both approaches high frequency components in the coupling signal (output of the subsystem) are filtered by an implemented anti-aliasing filter, according to the macro-step-size,³ and the mitigated frequency parts of the coupling signal are added by the NEPCE or EPCE scheme. This way, energy is preserved and the coupling challenge is relaxed. Figure 1.5 gives a compact overview about the approach. It illustrates (a) the zero-order extrapolated input signal $\hat{u}(t)$ incl. related discontinuities, (b) the excited stiff response $y(t)$ at the subsystem output and (c) the corrected solution $\tilde{y}(t)$ after application of the NEPCE anti-aliasing filter and the EPCE approach. Values at the coupling time instants are corrected w.r.t. energy-preservation, leading to an energy preserving mapping of high frequencies into low frequency ranges, based on *Parseval's identity*.

Figure 1.6 illustrates resulting spectra in non-logarithmic scale based on an ideal, uniform input spectrum of the input signal $y(t)$ in Eqs. 1.19–1.22, i.e. white noise. The upper and the mid plots show the resulting spectra after application of the NEPCE anti-aliasing filter and the CMA filter, respectively. Both plots depict the corresponding infinite spectra of the integrated, scaled and sampled input signals (*integrated sampled input*). For application of the NEPCE anti-aliasing filter $\xi = 2$ is chosen leading to an artificially step-size of $\Delta T/2$ (upper plot); CMA application uses the macro-step-size ΔT (mid). The NEPCE anti-aliasing filter (*NEPCE*

²Modelica library has velocity generators with build-in low pass filters to make it mathematically sound. However, these simple one- or two pole low pass filters dissipate energy. Physics-based low pass filters allow tuning such that energy loss can be controlled and preserved up to solver tolerances if required.

³It is worthy to point out that the macro-step size is the single tuning parameter of the energy-preserving anti-aliasing filters.

Faa) with pure linear interpolation leads to a moderate mitigation of higher frequency components, which is further improved by subsequent ZOH extrapolation (*NEPCE Faa+ZOH*). Obviously, the CMA filter ($\vartheta = 2$) with pure ZOH extrapolation possesses an increased amplification of higher frequencies due to the introduced discontinuities at coupling time instants. The increased phase shift of the CMA filter is motivated by averaging over two macro-time steps. Both approaches exhibit zero gain at the corresponding *Nyquist* frequencies.

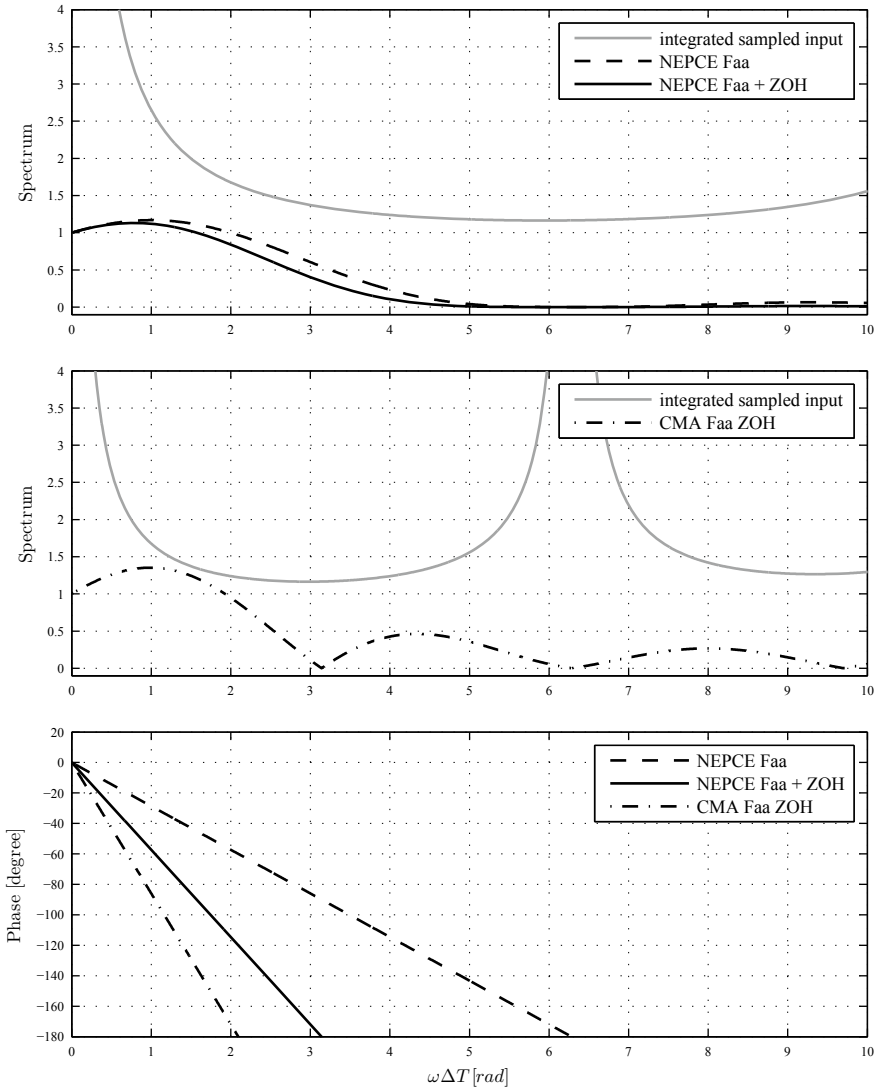


Fig. 1.6 Resulting energy-preserved anti-aliasing filtered and extrapolated spectra

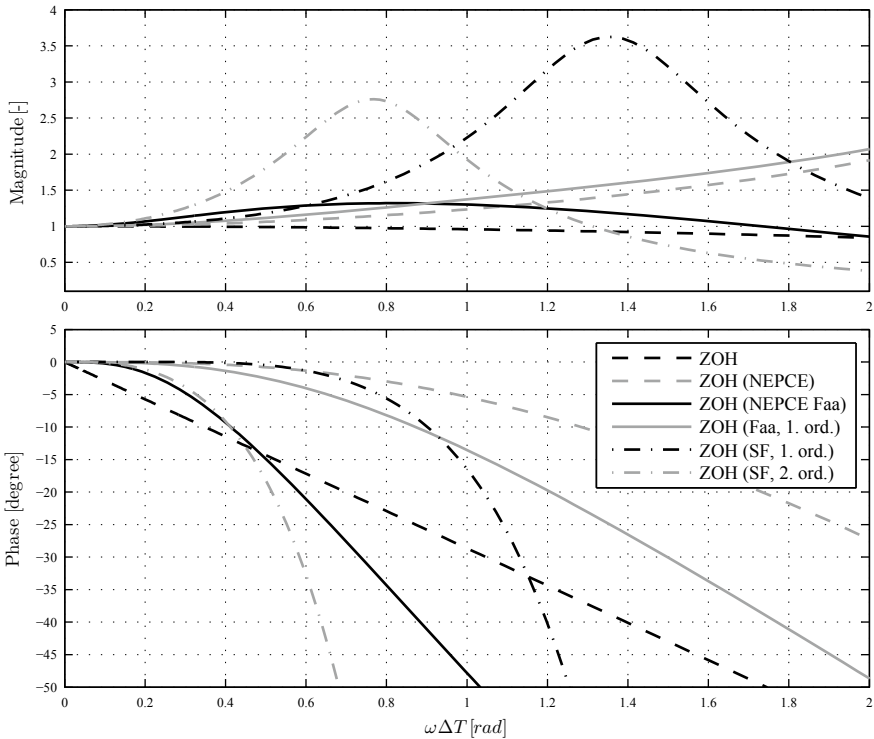


Fig. 1.7 Bode-Diagrams indicating the effect of the outlined energy-preserving coupling schemes for different configurations

Remark Depending on the chosen macro-step-size the high frequency components of the coupling signal are mapped in an energy preserving manner into low frequency ranges. This way the macro-step-size determines the bandwidth of the output signal and prevent aliasing effects. Furthermore, especially in the case of stiff (and extra-stiff) co-simulations the macro-step-size of an explicit co-simulation scheme has typically to be reduced significantly in order to ensure stability, which leads to an enormous and unacceptable increase of the overall co-simulation time. By applying the proposed approach the high-frequency parts within the output signal are eliminated, which enables the application of enlarged macro-step-sizes. As a consequence, stiffness is relaxed for application of (numerical) explicit coupling schemes motivating for introduction of the term *numerical stiffness*—finally, *numerical stiffness* is relaxed.

1.3.3 Handling Discontinuities

An intuitive solution to this problem is the application of additional low-pass filters to smooth the extrapolated coupling signal. Thereby, high frequency components are suppressed but in addition a filter inherent phase-shift is introduced. In closed-loop systems, which occurs naturally if extrapolations are necessary, this additional phase-shift affects the dynamic behavior of the co-simulated system. Another approach published in [25] suffers from the same problem. In fact, arrangements to mitigate high frequency components will improve solver accuracy but in contrast these methods are acting contra-productive concerning phase-shift. However, the concept presented above implements a smoothing filter as shown in Fig. 1.4. The mentioned adverse effects of the designed smoothing filter are interpreted as an additional co-simulation discretization error and is compensated by the compensation system.

Finally, Fig. 1.7 illustrates the transfer behavior of different configurations of the proposed coupling element via the *Bode'-Diagram*. The typically applied ZOH approach is drawn as reference. Application of the NEPCE concept without additional filters leads to an enhanced *efficient bandwidth* ($\gamma = 100$, [28]). Otherwise, the compensation of implemented anti-aliasing and smoothing filters is depicted. On the one hand, the phase shift introduced by the embedded NEPCE anti-aliasing filter is compensated within a specific lower *efficient bandwidth*. Due to the cut-off frequency of the classical anti-aliasing filter of first order at $\omega\Delta T = 1$ rad, the *efficient bandwidth* of the classical anti-aliasing filter (*Faa, 1. ord*) outperforms the NEPCE anti-aliasing filter utilization; the pure NEPCE anti-aliasing filter behaves like a time delay of $\Delta T/2$. On the other hand, the compensation performance using a smoothing filter of first order is superior to the compensation performance by using a classical anti-aliasing filter of first order, as the smoothing filter is considered within the synthesis of the related compensation system. Furthermore, the order of the applied smoothing filter significantly degrades the performance (i.e. the *efficient bandwidth*) of the coupling element. However, in all cases the transfer behavior of the coupling elements is superior to the uncorrected ZOH coupling approach for the lower frequency range.

1.4 Theoretical Example: Lumped Propeller Shaft

1.4.1 Lumped Propeller Shaft

A propeller shaft is arbitrarily chosen to consist of lumped inertia's connected by spring-damper elements and then divided over 5 co-simulation subsystems as shown in Fig. 1.8. It represents a benchmark example [32] and shall demonstrate the basic energy-preserving concepts, but also represents a realistic challenge when it comes to couplings in rotational mechanics, e.g. half-shafts of a car. Parameters are chosen

as follows: inertia's $J_1 = 0.9 \text{ kgm}^2$, $J_2 = 0.7 \text{ kgm}^2$, $J_3 = 0.5 \text{ kgm}^2$, $J_4 = 0.3 \text{ kgm}^2$, $J_5 = 0.1 \text{ kgm}^2$, damping coefficients $c_s = 1000 \text{ Nm/rad}$ and damping coefficients $c_d = 44.27 \text{ kg/s}$.

For reference, the *exact* results are created with an equivalent model without the sampling effects. The propeller shaft is accelerated with a torque at $t = 1.0 \text{ s}$ and after a set time at $t = 2.0 \text{ s}$ a counter acting torque of the same amount is applied at the other end of the propeller shaft. Until the counter acting torque is applied, the shaft will accelerate and after the counter acting torque is applied a constant steady-state rotational velocity of $\omega = 4 \text{ rad/s}$ is reached.

1.4.2 Lumped Propeller Shaft with Clutch

To further exercise the methodology, a clutch is introduced. The clutch is deployed from the Modelica Standard Library and is activated with a step function at $t = 1.5 \text{ s}$. For mathematical reasons, the clutch needs to be inserted in between two inertia's where FMU 3 was replaced, see [32] for further details.

1.4.3 Evaluation Results

This paper will only present example results and is not meant to be exhaustive. Furthermore, for illustration purposes only the angular velocity of the fifth element (FMU 5) of the rotational cascades is plotted.

Macro-step-size 10 ms—First co-simulations are carried out at 10 ms macro-step-size, see Fig. 1.9 (top). The result from a pure ZOH co-simulation is shown for comparison purposes. Furthermore, co-simulations with application of the energy-

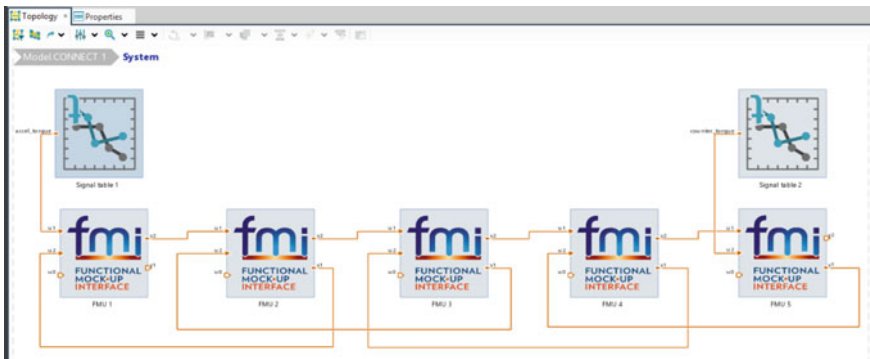


Fig. 1.8 Benchmark model of lumped propeller shaft in AVL Model.CONNECT™. The propeller shaft consists of 5 FMUs

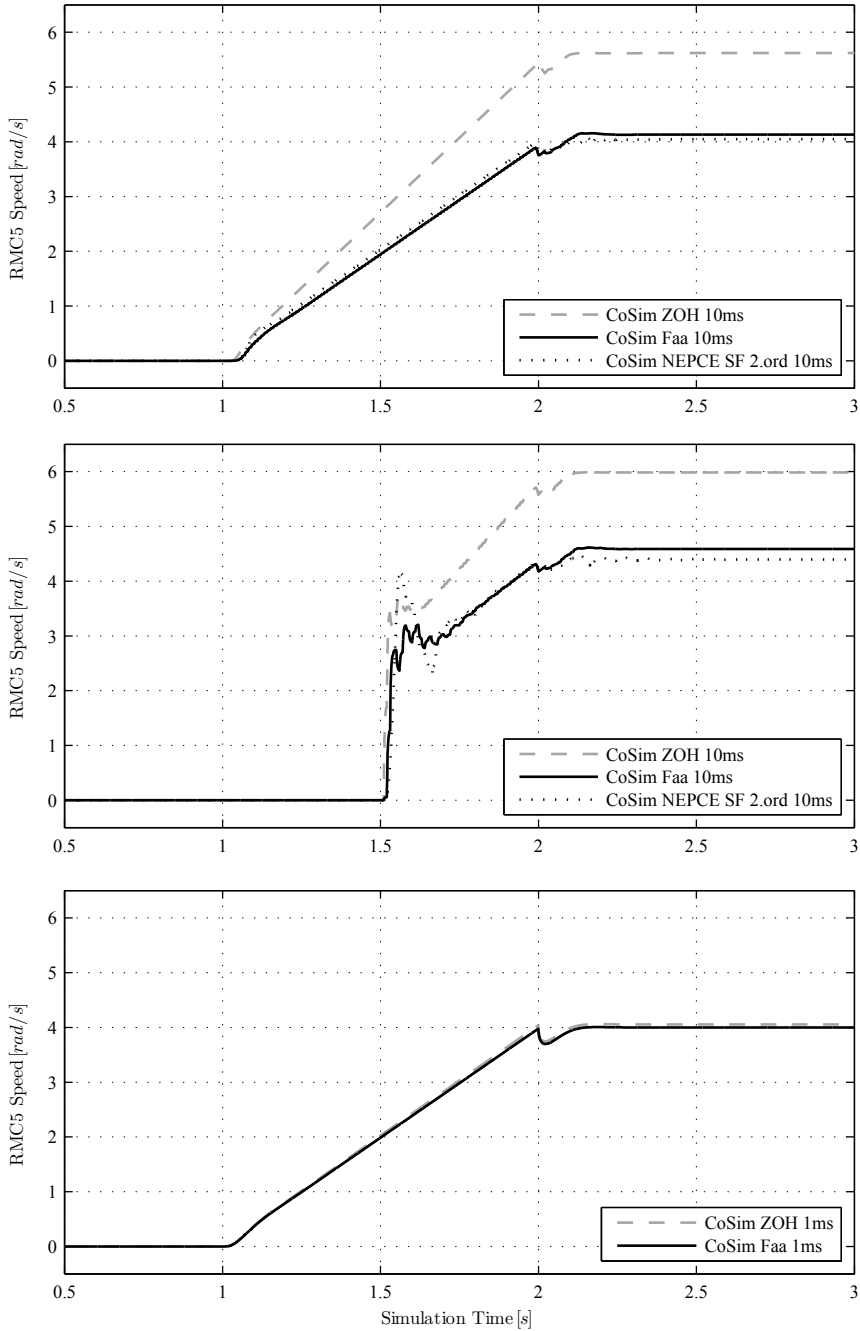


Fig. 1.9 Lumped propeller shaft results for $\Delta T = 10$ ms (top), with clutch for $\Delta T = 10$ ms (mid) and lumped propeller shaft without clutch $\Delta T = 1$ ms (bottom)

preserving anti-aliasing filter with ZOH extrapolation and with application of a smoothing filter of second order were conducted. The spring-damper settings causes design frequency of 5 kHz, well beyond the Nyquist frequency of 50 Hz. The resulting angular velocities of the cascaded inertia's utilizing energy-preserving concepts show clearly the energy preservation by means of acceleration levels being equal. The final steady-state rotational velocities however, are different from the exact results. This is solely caused by the remaining co-simulation discretization error and the resulting introduced time delays. The non-smooth activation at $t = 1.5$ s of the clutch, Fig. 1.9 (mid), does not cause any stability issues. The transients visible are caused by the bandwidth of the overall systems sampling frequency. Slightly improved steady state rotational velocities are achieved by the smoothing filter application, as the effect of the extrapolation and the smoothing filter, i.e. the co-simulation discretization error, is compensated and no fast dynamics are excited within the subsystems. Otherwise, increased oscillations are introduced by the smoothing filter approach indicating stability limitations.

Macro-step-size 1 ms—To exemplify some of the above made statements, the macro-step-size is reduced to 1 ms and the co-simulations with application of the pure ZOH extrapolation and the energy-preserving anti-aliasing filter were repeated. Again, the acceleration levels are correct and the final velocity difference has reduced, please compare the top and bottom plots within Fig. 1.9. The overall bandwidth with a smaller macro-step-size becomes higher.

1.5 Industrial Example: Electric Machine FOC

The lumped propeller shaft benchmark has been solely created for exploring the non-iterative co-simulation challenges and the demonstration of the outlined energy-preserving concepts. The electric motor with its field-oriented control using high frequency switching inverters is a realistic challenge and an industrial example, representing a further co-simulation benchmark [33]. This switching will cause high frequency contents in the electric torque generated in the motor. For system simulation the frequency content of the inverter is not of interest, but the field oriented control is important for its efficient control of currents.

1.5.1 The Model

The electric motor model and its inverters is based on standard Modelica models. The models are entirely built with first principle physics as depicted in Fig. 1.10. No mappings nor any other system identification techniques are deployed. The inverter switches at design frequency and in this particular case it is set to 10 kHz. This frequency is significantly higher (!) than the bandwidth of interest at systems level; e.g. the mechanical domain.

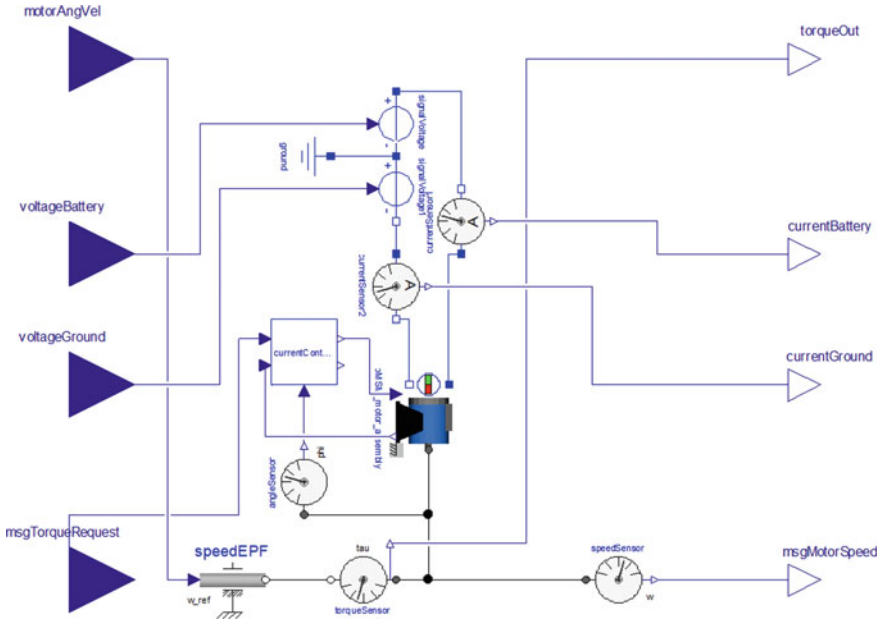


Fig. 1.10 Modelica model of a PMSM with FOC control with interface for co-simulation export

The load model is simply represented by an inertia and a quadratic resistance. The inertia of the load shall be larger than the internal inertia of the electric machine, for reasons of system stability. The resistive torque load is a function of rotational velocity squared, which resembles a load for traction purposes.

1.5.2 Evaluation Results

Four different simulations are compared in this study. A monolithic solution done within Dymola serving as reference and deemed as the true solution. Secondly, the prevailing non-iterative co-simulation solution, i.e. pure zero-order-hold extrapolation. For simulations three and four the NEPCE energy-preserving anti-aliasing filter with ZOH extrapolation and the EPCE approach are applied, respectively, the inputs to the FOC subsystem are extrapolated by ZOH. Simulations are performed with constant macro-step-sizes of 10 and 1 ms and Fig. 1.11 illustrates the results.

The simulation results for the four alternative approaches show some differences. First of all the ZOH alternative accelerates significantly faster than the other three. Irrespectively of the used macro-step-size, the transient response of the ZOH approach is significantly off from the reference traces and aliasing effects are recognizable by the oscillations in steady-state. Furthermore, the solutions by application of the NEPCE anti-aliasing filter as well as of the EPCE approach shows moder-

ate deviations from the monolithic solution. By reducing the macro-step-size (2nd row) the EPCE results almost fits to the reference whereas the deviation of the NECPE solution slightly increases. The difference is caused by the kind of implementation of the energy-preserving anti-aliasing filters. In case of EPCE application the energy-preserving anti-aliasing filter is implemented within the FOC subsystem (model-based solution). On contrary, the NEPCPE energy-preserving anti-aliasing filter is implemented within the co-simulation platform at couplings between subsystems. As both filters map high-frequency contents into lower ranges in an energy-preserving manner the results should match. No oscillations in steady-state are recognizable. The reason for the deviation is quite simple. EPCE is implemented within the simulation model and all information, like micro-steps, is utilized. On the other hand, FMU's are integrated and outputs values are exclusively available at coupling time instants. As work-around the FMU's are sub-sampled at a significantly lower step-size for gathering the internal subsystem behavior, whereby the selection of the sub-sampling step-size is challenging. The 3rd and 4th plots illustrated the torques of both approaches respectively. Additional torque peaks are introduced due to discontinuities at the FOC subsystem inputs, where in case of subsampling more discontinuities are introduced. In particular the case of 1 ms, compared to the 10 ms co-simulation, more torque peaks are introduced at coupling time instants and more information is lost due to mandatory subsampling. Energy is artificially produced within the coupling resulting in a slightly increased acceleration of the electric machine. This example indicates a current limitation of the FMI standard. A possible solution could be the exchange of all *intermediate output values* over the macro-time step at the coupling time instant. This way, both solutions would lead to almost equal results.

Remark From an abstract point of view during co-simulation the subsystem models are solved by tailored solvers and the individual output signals are then sampled by the chosen macro-step-size. As demonstrated within this contribution, problems arise when the macro-step-size does not fit to the corresponding subsystem dynamics. Aliasing effects occur and the basic subsystem behavior is obfuscated. These motivates for the exchange of eventually available *intermediate output values*, which is currently under discussion as an extension to the current FMI standard.

Another option is represented by implementation of energy-preserving, continuous time moving-average filters at outputs and signal reconstruction filters at inputs within the individual subsystem models. In contrast to the herein discussed *co-simulation platform-based approach*, this would lead to a *co-simulation model-based approach*, which requires the extension of the individual subsystem. These concept as well as a related co-simulation methodology is presented and discussed in [31].

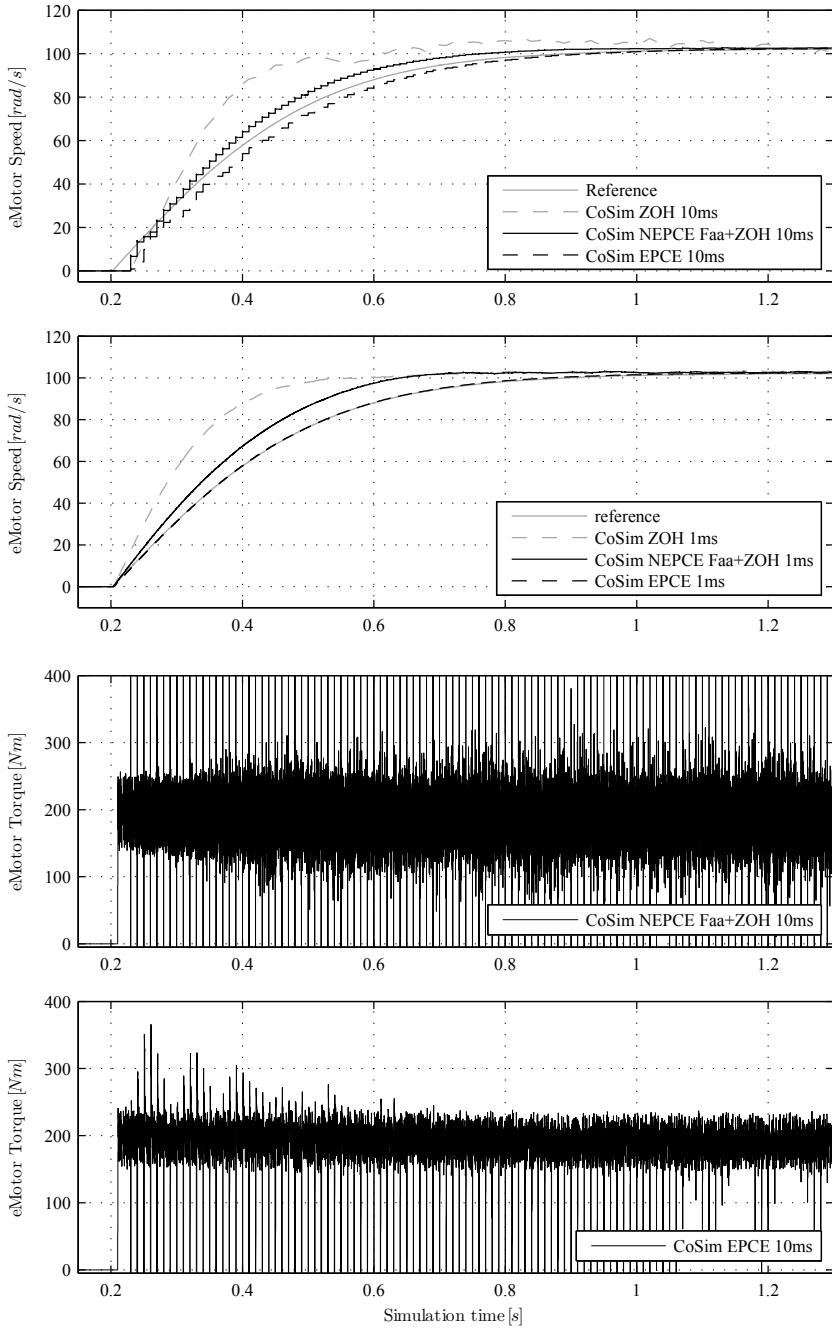


Fig. 1.11 Electric Machine FOC results for $\Delta T = 10$ ms and $\Delta T = 1$ ms for different coupling configurations; shaft speeds (upper); torques (lower)

1.6 Conclusion

This contribution investigates non-iterative co-simulation from an system-oriented point of view and revises different recently proposed energy-preserving approaches. Besides identification of the three non-iterative co-simulation challenges, a special focus is put on *numerical stiffness* in terms of sampling frequency versus time constants of the entire system dynamics. In case of aliasing high frequency components of the coupling signal are mapped into lower frequency ranges based on *Parseval's* identity. The NEPCE and EPCE technology are control engineering wise the same solutions, but applied from two different perspectives. The NEPCE is a co-simulation platform-based approach, whereas the EPCE is co-simulation model-based approach. Identified limitations of the FMI standard shall encourage further FMI developments.

Remark The presented work describes novel approaches for energy-preserved coupling in non-iterative co-simulation and weak-coupled problems. Protected by a granted European patent (EP-2-442-248-B1) [34] the outlined schemes are implemented within the co-simulation platform AVL Model.CONNECTTM [1]. Furthermore, another European patent application (EP-3-136-267-A1) [35] intends to protect the EPCE approach.

Acknowledgements This work was accomplished at the VIRTUAL VEHICLE Research Center in Graz, Austria and at the Volvo Car Corporation in Göteborg, Sweden. The authors would like to acknowledge the financial support of the COMET K2—Competence Centers for Excellent Technologies Programme of the Austrian Federal Ministry for Transport, Innovation and Technology (bmvit), the Austrian Federal Ministry of Science, Research and Economy (bmfw), the Austrian Research Promotion Agency (FFG), the Province of Styria and the Styrian Business Promotion Agency (SFG) and the Swedish Energy Agency (Energimyndigheten).

References

1. AVL Model.CONNECTTM, The neutral model integration and co-simulation platform connecting virtual and real components. www.avl.com/-/model-connect (2018). Accessed 31 Jan 2018
2. CosiMate, Heterogeneous co-simulation in distributed environments. <https://site.cosimate.com> (2018). Accessed 31 Jan 2018
3. xMODTM, Multi-model integration and virtual experimentation. www.xmodsoftware.com (2018). Accessed 31 Jan 2018
4. Functional Mock-up Interface (FMI), A tool independent standard to support both model exchange and co-simulation of dynamic models using a combination of xml-files and compiled C-code. www.fmi-standard.org (2018). Accessed 31 Jan 2018
5. Blochwitz, T. et al.: Functional mockup interface 2.0: the standard for tool independent exchange of simulation models. In: Proceedings of the 9th International Modelica Conference, Munich, Germany (2012)
6. Kübler, R., Schiehlen, W.: Two methods of simulator coupling. *Math. Comput. Model. Dyn. Syst.* **6**, 93–113 (2000)
7. Busch, M.: Zur effizienten Kopplung von Simulations programmen. Dissertation, Kassel University (2012)

8. Gomes, C. et al.: Co-simulation: state of the art. Cornell University Library (2017). [arXiv:1702.00686](https://arxiv.org/abs/1702.00686)
9. Kübler, R., Schiehlen, W.: Modular simulation in multibody system dynamics. *Multibody Sys. Dyn.* **4**, 107–127 (2000)
10. Arnold, M., Schiehlen, W. (eds.): *Simulation Techniques for Applied Dynamics*. CISM Course and Lectures, vol. 507. Springer, Vienna (2009)
11. Sun, J., Grotstollen, H.: Fast time-domain simulation by waveform relaxation methods. In: *27th Annual IEEE Power Electronics Specialists Conference* (1996)
12. Hairer, E., Wanner, G.: *Solving Ordinary Differential Equations Stiff and Differential-Algebraic Problems*. Springer Series in Computational Mathematics, 2nd edn. Springer, Berlin (2010)
13. Arnold, M., Burgermeister, B., Eichberger, A.: Linearly implicit time integration methods in real-time applications: DAEs and stiff ODEs. *Multibody Syst. Dyn.* **17**(2), 99–117 (2007)
14. Burgermeister, B., Arnold, M., Esterl, B.: DAE time integration for real-time applications in multi-body dynamics. *ZAMM, Z. Angew. Math. Mech.* **86**(10), 759–771 (2006)
15. Sicklinger, S., et al.: Interface Jacobian-based co-simulation. *Int. J. Numer. Methods Eng.* **98**(6), 418–444 (2014)
16. Elmqvist, H., Olsson, H., Belsky, V., Engelmann, B.: *Co-simulation Procedures Using Full Derivatives Of Output Variables*. Dassault Systmes, EP2680157A1 (2014)
17. Viel, A.: Implementing stabilized co-simulation of strongly coupled systems using the Functional Mock-up Interface 2.0. In: *Proceedings of the 10th International Modelica Conference*, Lund, Sweden (2014)
18. Sadjina, S., Pedersen, E.: *Energy Conservation and Coupling Error Reduction in Non-iterative Co-simulations*. Cornell University Library (2016). [arXiv:1606.05168](https://arxiv.org/abs/1606.05168)
19. Sadjina, S., Kyllingstad, L.T., Pedersen, E., Skjong, S.: *Energy Conservation and Power Bonds in Co-simulations: Non-iterative Adaptive Step Size Control and Error Estimation*. Computing Research Repository (2016). [arXiv:1602.06434](https://arxiv.org/abs/1602.06434)
20. Braun, R., Krus, P.: *Tool-Independent Distributed Simulations Using Transmission Line Elements And The Functional Mock-up Interface*. Linköping University, Division of Fluid and Mechatronic Systems, SE-58183 Linköping, Sweden (2013)
21. Fernández, J., Kofman, E.: A stand-alone quantized state system solver for continuous system simulation. *J. Simul. Soc. Comput. Simul. Int.* **90**(7), 782–799 (2014)
22. Andersson, C.: *A software framework for implementation and evaluation of co-simulation algorithms*. Lund University (2013)
23. Benedikt, M., Watzenig, D., Hofer, A.: Modelling and analysis of the non-iterative coupling process for co-simulation. *Math. Comp. Model. Dyn.* **9**(6), 451–470 (2013)
24. Benedikt, M., Watzenig, D., Zehetner, J., Hofer, A.: NEPCE A nearly energy-preserving coupling element for weak-coupled problems and co-simulations. *ECCOMAS Computational Methods for Coupled Problems in Science and Engineering V*, Ibiza (2013)
25. Kossel, R., Strupp, N. and Tegethoff, W.: Effects of tool coupling on transient simulation of a mobile air-conditioning cycle. In: *Proceedings of the 7th International Modelica Conference*, pp. 318–325. The Modelica Association (2009)
26. Janschek, K.: *Mechatronic Systems Design: Methods, models, Concepts*. vol. XXI, 805 p. (2012). ISBN: 978-3-642-17530-5
27. Kraft, J., Meyer, T., Schweizer, B.: Reduction of computation time by parallelization and incorporating co-simulation techniques. In: *IUTAM Symposium on Co-simulation and Solver-Coupling*, Darmstadt, Germany (2017). (to be published)
28. Benedikt, M., Hofer, A.: Guidelines for the application of a coupling method for non-iterative co-simulation. In: *IEEE, 8th Congress on Modelling and Simulation (EUROSIM)*, Cardiff, Wales (2013)
29. Benedikt, M.: *A coupling methodology for non-iterative co-simulation*. Ph.D. Thesis, Graz University of Technology (2012)
30. Oppenheim, A., Schaffer, R., Buck, J.: *Discrete Time Signal Processing*, 2nd edn. Prentice Hall, Upper Saddle River (1999)

31. Drenth, E.: Robust co-simulation methodology of physical systems. In: 9th Graz Symposium Virtual Vehicle (2016)
32. Drenth, E.: VVA20160303 VVA Co-Simulation Benchmark: Lumped Propeller Shaft. Volvo Car Corporation (2016)
33. Drenth, E.: VVA20170302 VVA Co-Simulation Benchmark: FOC Electric Machine Co-Simulation. Volvo Car Corporation (2016)
34. Benedikt, M., Watzenig, D., Bernasch, J.: Coupling method for non-iterative co-simulation. Das virtuelle Fahrzeug, EP2442248(B1) (2010)
35. Drenth, E.: Method and System for Control and Co-Simulation of Physical Systems. Volvo Car Corporation, EP3136267(A1) (2015)

Chapter 2

TLM-Based Asynchronous Co-simulation with the Functional Mockup Interface



Robert Braun, Robert Hällqvist and Dag Fritzon

Abstract Numerical stability is a key aspect in co-simulation of physical systems. Decoupling a system into independent sub-models will introduce time delays on interface variables. By utilizing physical time delays for decoupling, affecting the numerical stability can be avoided. This requires interpolation, to allow solvers to request input variables for the time slot where they are needed. The FMI for co-simulation standard does not support fine-grained interpolation using interpolation tables. Here, various modifications to the FMI standard are suggested for improved handling of interpolation. Mechanical and thermodynamic models are used to demonstrate the need for interpolation, as well as to provide an industrial context. It is shown that the suggested improvements are able to stabilize the otherwise unstable connections.

2.1 Introduction

Numerical robustness is a key factor in simulation solver coupling. Using different solvers for different parts of a simulation model can be of great benefit in terms of increasing simulation speed and robustness. However, all variables shared by more than one solver will need to be delayed in time. This may result in numerical errors and instability. One solution is to decouple the model where significant physically motivated time delays are present. In this way, all time delays are a natural part of the model and no non-physical time delays will thus need to be inserted.

R. Braun (✉)

Division of Fluid and Mechatronic Systems, Linköping University, Linköping, Sweden
e-mail: robert.braun@liu.se

R. Hällqvist

Department of Systems Simulation and Concept Analysis, Saab Aeronautics,
Stockholm, Sweden
e-mail: robert.hallqvist@saabgroup.com

D. Fritzon

SKF Group Technology, AB SKF, Gothenburg, Sweden
e-mail: dag.fritzon@skf.com

© Springer Nature Switzerland AG 2019

B. Schweizer (ed.), *IUTAM Symposium on Solver-Coupling and Co-Simulation*,
IUTAM Bookseries 35, https://doi.org/10.1007/978-3-030-14883-6_2

The Functional Mock-up Interface (FMI) is a tool-independent standardized interface for connecting simulation models from different modelling and simulation tools [3]. A model is exported as a Functional Mock-up Unit (FMU), containing executable C code and an XML description file. Co-simulation is conducted by importing multiple FMUs to a master simulation tool (MST) and connecting them in a *composite model*. FMUs can be either for co-simulation (FMI CS) or for model exchange (FMI ME). With FMI CS, each FMU has its own internal solver and exchanges data only at pre-defined communication points. This enables discrete-time co-simulation, where each FMU internally may use continuous-time simulation. With FMI ME, the MST must provide a solver. Derivatives of state variables can then be updated at any time, which enables continuous-time co-simulation. Experiments have been conducted to investigate how the FMI standard can be combined with physically motivated model decoupling. Four different methods for extrapolation and interpolation of interface variables have been implemented and compared. Based on the results, some improvements to the FMI standard are suggested.

This paper uses a domain-independent notation of the exchanged variables. All physical connections use *intensity* and *flow* variables. For the mechanical and fluid domains, intensity would equal force and pressure while flow would equal velocity and volume flow.

2.1.1 Problem Description

Transmission Line Modelling (TLM) is a well-known technique for decoupling of simulation models [11]. The basic idea is that in reality information propagation speed is always limited. This includes, for example, stress waves in materials or pressure waves in fluids. Hence, every physical element has a natural time delay. By including physical time delays in model variables, equations can be separated without affecting numerical stability. A TLM element and its equations are shown in Fig. 2.1. F is the force, v velocity, Δt the time delay and Z_c characteristic impedance. The delayed information traversing the element is denoted the *wave variable*. A co-simulation MST using TLM has been developed by SKF [14]. The implementation is based on asynchronous socket communication. Each slave tool has fully independent time variables and step sizes. Due to the physical time delays, input data is available not only at the beginning of the step but also during the step via interpolation. Slave tools use callback functions for receiving inputs and sending outputs for specific

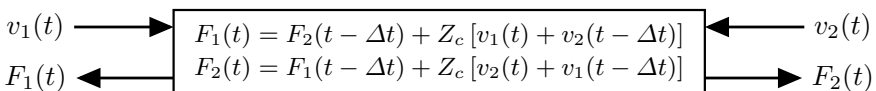


Fig. 2.1 A TLM element with its equations

time instances. Input variables are thus kept up-to-date even during internal iterations performed by the slave.

The FMI standard aims to reduce overhead costs associated with simulation tool coupling by means of a standardized interface. FMI is stipulated to enhance and simplify model export and integration in both industry and academia. It is therefore desirable to make the MST compatible to the FMI standard. This would enable simulation of multiple connected Functional Mock-up Units (FMUs) exported from any simulation tool with FMI support. While FMI for model exchange can easily be adapted for the TLM-based co-simulation MST, FMI for co-simulation induces several challenges concerning numerical stability. Since input variables can only be updated at the beginning of each communication step, the stability benefits of TLM are lost.

2.1.2 Related Work

A prototype of a master simulation tool for FMI-based co-simulation was presented in [2]. Master algorithms combining FMI-based co-simulation with the High-Level Architecture (HLA) standard have also been developed [1, 8, 12]. A TLM-based co-simulation MST with FMI support was implemented in the Hopsan simulation tool [5]. In [4] it was shown that it is possible to connect two simulation tools with a TLM element, using FMI, without a master simulation tool orchestrating the simulation. The MST used in this paper differs from the previous experiments in that it uses asynchronous data communication. Consequently, each TLM connection can have its own independent time delay, and each FMU can use its own independent simulation step size. With synchronous communication, all connections must have the same time delay and each sub-model must be able to provide output variables at this interval.

Solving two parts of a simulation model simultaneously on different solvers without using physical time delays requires incorporating numerical time delays, which may affect accuracy and numerical stability. Errors can usually be reduced by reducing the size of the delays, at the cost of a longer simulation time. One solution to reduce the resulting negative impact on simulation time is to use adaptive communication step-size [13]. In contrast to physically motivated decoupling, this method requires the FMU to support setting and getting FMU states.

2.2 Numerical Solutions

Several solutions, both with and without modifications to the FMI standard, are implemented and analysed. Three different variable estimation methods are used, see Fig. 2.2. With constant extrapolation, variables are kept constant during each communication step. Coarse-grained interpolation means that the value at the beginning

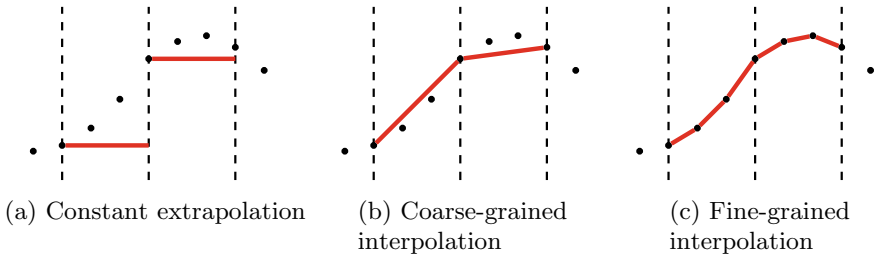


Fig. 2.2 Three variable estimation methods are used to stabilize the connections

and at the end of the step, i.e. at the communication points, are used for interpolation. Fine-grained interpolation includes the values between communication points as well. Experiments are limited to linear interpolation. Higher order interpolation methods may be used to improve accuracy further, but has not been considered further in this paper.

2.2.1 Constant Extrapolation

The simplest method for exchanging variables is to use constant extrapolation, see Fig. 2.2a. Input variables are updated at the beginning of each step and remain constant during the step. This solution is fully supported by the current FMI for co-simulation standard. However, numerical stability cannot be guaranteed. If the FMU supports saving and loading FMU states, it is possible to improve stability by using adaptive communication step size [13]. Unfortunately, this feature is optional in the standard and many exporting tools do not support it. Stability can then only be achieved by using a sufficiently small constant communication step size during the entire simulation. This induces a severe performance penalty. Furthermore, it is not possible to tell in advance whether or not the step size is small enough.

2.2.2 Coarse-Grained Interpolation

It is possible to provide an FMU with the approximated values of input variable time derivatives. These can be used by the FMU for interpolation or extrapolation, see Fig. 2.2b. For a delayed variable the value is known both at the beginning and at the end of the step. Hence, the first-order derivative can easily be approximated. The resolution of the interpolation, however, will be limited since all data in the interpolation table will not be used. Furthermore, the delayed information will consist of the wave variable and not the intensity variable. The FMU must therefore include the TLM boundary equations, and compute the intensity internally. While techni-

cally possible, it makes it harder to generalize the method for FMUs from arbitrary generation tools.

2.2.3 *Fine-Grained Interpolation Inside the FMU*

Here, fine-grained interpolation means that all points in the interpolation table are used, as shown in Fig. 2.2c. The actual interpolation can either be performed inside the FMU or provided by the master simulation tool from callback functions.

Interpolation inside the FMU with high resolution is possible only if the FMU is provided with the interpolation table. Interpolation data can be sent as normal variables, using the `fmi2SetReal()` function. However, this requires customized FMUs and manual adjustments will therefore be required, and it will not work for all simulation tools.

Furthermore, it requires a large amount of data exchange, which may affect simulation performance. If support for populating interpolation tables in FMUs could be incorporated into the standard, this could become a more general and computationally efficient solution. A proposed function for setting time-stamped variables is shown in Listing 2.1 Like the coarse-grained interpolation, this approach also requires the FMU to include the TLM equations.

Listing 2.1 A proposal for extending the FMI API to support time-stamped variables

```
fmi2Status fmi2SetReal (fmi2Component c,
                       const fmi2ValueReference vr[],
                       size_t nvr,
                       const fmi2Real value[],
                       const fmi2Real time[]);
```

Callback functions for reading and writing input variables at specified times could be provided to the FMUs. Interpolation can then be handled by the MST. This would preserve the guaranteed stability provided by TLM, while exposing only a minimal interface consisting of intensity and flow variables. The exported models would not need any adaption for TLM. However, this method requires an extension to the current FMI standard. A proposal for such an extension is shown in Listing 2.2

Listing 2.2 A proposal for extending the FMI API with callback functions for requesting interpolated inputs

```
typedef fmi2Real (*fmi2GetRealCb) (fmi2ValueReference,
                                   fmi2Real,
                                   fmi2Real);

fmi2Status fmi2SetGetRealCb(fmi2Component c,
                             const fmi2GetRealCb cb);
```

An alternative solution to callback functions is to use pure discrete-event simulation using FMI for model exchange. The slave can have its own custom solver internally,

without exposing any continuous state variables to the MST. In this way, FMI ME can be used for co-simulation with distributed solvers, in contrast to FMI ME for continuous-time simulation, where the MST must provide the solver. Whenever an interpolated input variable is required or an output variable is available, the slave informs the MST by using time events. With the current API, however, it is not possible to distinguish input data events from output events. The API would thus have to be extended with an additional flag in the `fmi2EventInfo` structure, as shown in Listing 2.3

Listing 2.3 A proposal for extending the FMI event info class for events with only inputs

```
typedef struct {
    fmi2Boolean  newDiscreteStatesNeeded;
    fmi2Boolean  terminateSimulation;
    fmi2Boolean  nominalsOfContinuousStatesChanged;
    fmi2Boolean  valuesOfContinuousStatesChanged;
    fmi2Boolean  outputsNotAvailable;
    fmi2Boolean  nextEventTimeDefined;
    fmi2Real     nextEventTime;
} fmi2EventInfo;
```

2.2.4 One-Step Functions

With FMI for co-simulation, each FMU writes output variables only after every completed communication step. Interpolation accuracy can be improved by letting each FMU provide output variables as often as possible. The FMI standard provides a function called `fmi2doStep()`, which tells the slave to simulate to a specified stop time. If the MST could tell the FMU to take a step without specifying a stop time, the FMU could simulate until the next time it is able to provide output data and then return the actual stop time to the MST. This would make it possible for the MST to obtain output data as often as possible, which would enable more densely populated interpolation tables. Many numerical solvers, such as CVODE and IDA, already provide one-step execution [10]. It is, however, unclear how this function would work for FMUs not based on differential equations.

2.3 Numerical Experiments

Four example composite models are used to illustrate the need for and to verify the feasibility of the proposed solutions. The first model is a one-dimensional spring-mass system. The second two models are two-dimensional pendulums. 2D mechanics models are usually more stiff and put stricter requirements on numerical stability. Finally, a thermodynamic connection extracted from an industrial use case is investigated.

The first three examples are implemented using custom FMUs, written in plain C++. The FMUs support first order input derivatives for coarse-grained interpolation. Interpolation tables are provided to FMUs by repeatedly calling the standard `fmi2setReal()` function. The FMI API is extended to support a prototype of callback functions. Discrete event simulation is tested by (incorrectly) using the `valuesOfContinuousStatesChanged` flag to indicate whether output variables are available or not. Finally, the thermodynamic model is used to compare simulations implementing constant extrapolation, coarse-grained and fine-grained interpolation using FMUs exported from the commercial Modelica tool Dymola. Interpolation is implemented as Modelica code. A function for coarse-grained interpolation is shown in Listing 2.4 This makes it possible to use input derivatives even if the FMU does not support the native FMI function for this. Fine-grained interpolation is performed by an interpolation model and two functions, see Listings 2.5–2.7 The FMU is provided with wave variables and corresponding time variables. Then, the actual value is obtained by using the built-in Modelica function `interpolate(t, c, tc)`.

Listing 2.4 A Modelica function for interpolating input variables in Dymola using first order time derivatives

```
function LinearInterpolation
  input Real t "Time variable";
  input Real c "Wave variable";
  input Real dCdt "Time derivative of wave variable";
  input Real tc "Current time";
  output Real ci "Interpolated wave variable";

algorithm
  ci := c + dCdt * (tc - t);

end LinearInterpolation;
```

Listing 2.5 A Modelica model for interpolating input variables in Dymola

```
model Interpolate
  Modelica.Blocks.Interfaces.RealVectorInput c1[n];
  Modelica.Blocks.Interfaces.RealVectorInput t1[n];
  Modelica.Blocks.Interfaces.RealOutput pi;
  Modelica.Blocks.Interfaces.RealInput q1;
  Modelica.Blocks.Interfaces.RealInput Zc;
  parameter Integer n;

protected
  Real ci;

equation
  ci = PopulateInterpolate(t1, c1, time);
  pi = getPressure(ci, Zc, q1);

end Interpolate;
```

Listing 2.6 A Modelica function for interpolating the wave variable in Dymola

```

function PopulateInterpolate
  input Real t[:] "Time variable vector";
  input Real c[:] "Wave variable vector";
  input Real tc "Current time";
  output Real ci "Interpolated wave variable";

algorithm
  ci:=Modelica.Math.Vectors.interpolate(t,c,tc);

end PopulateInterpolate;

```

Listing 2.7 A Modelica function for requesting pressure variable at a specific time in Dymola

```

function getPressure
  input Real c "Wave variable";
  input Real q "Volume flow";
  input Real Zc "Characteristic impedance";
  output Real p "Pressure";

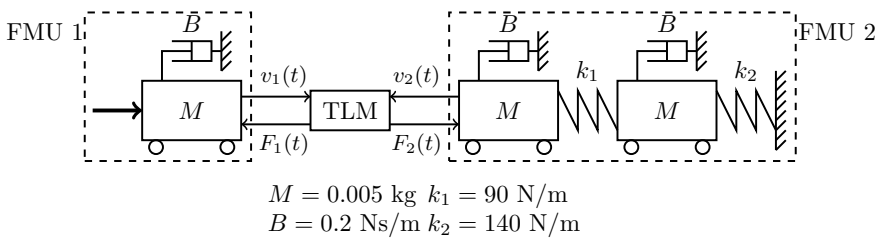
algorithm
  p:=Zc*q+c;

end getPressure;

```

2.3.1 1D Three-mass System

An example composite model consisting of a three-mass system separated into two FMUs is presented in Fig. 2.3. Simulation results verify the proposed method with second order dynamics in one dimension. One of the FMUs has internal dynamics with two different resonance frequencies. The composite model is simulated with a communication step size (i.e. TLM time delay) of 0.4 ms. After 0.1 s, a step force of 100 N is applied on the first mass. Parameters are intentionally chosen to make the simulation unstable when using constant extrapolation. Simulation results using constant extrapolation, coarse-grained interpolation, interpolation and callback func-

**Fig. 2.3** A test model consisting of a three-mass system is divided into two FMUs

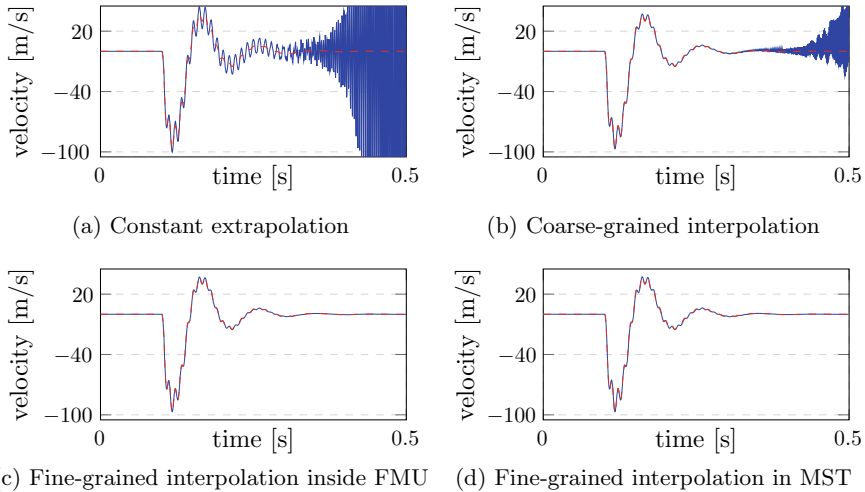
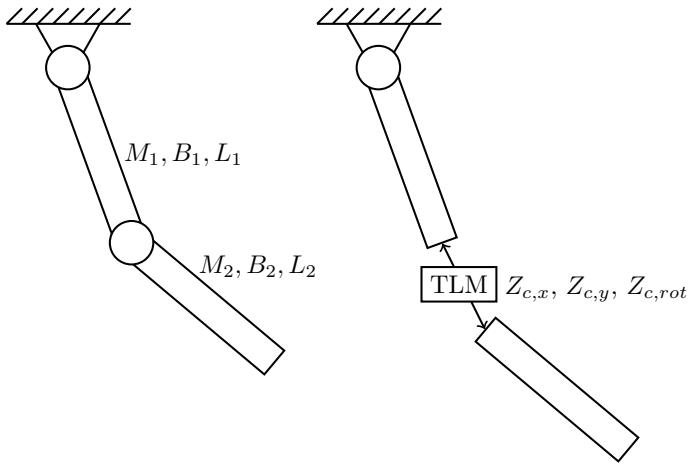


Fig. 2.4 Velocity against time at the left side of the TLM connection. Red dashed line is the exact reference solution

tions are shown in Fig. 2.4. Constant extrapolation and coarse-grained interpolation methods are unstable. Fine-grained interpolation is able to stabilize the coupling both when computed in MST and locally inside the FMU.

2.3.2 2D Double Pendulum

The second example composite model consists of a two-dimensional double pendulum, see Fig. 2.5. Double pendulums exhibit chaotic motion and are sensitive to initial conditions. This makes it an interesting example for verifying simulation techniques. The two arms are connected through a TLM element with a time delay of $\Delta t = 1e - 3$ s and a characteristic impedance of $Z_c = 1e5$ for X and Y directions. This corresponds to a spring of stiffness $K_s = 1e8$ N/m. Rotational impedance is set to zero. Hence, the TLM element represents a revolute joint with some flexibility. Simulation is initiated with both arms pointing horizontally. Results from simulations implementing the four different methods are shown in Fig. 2.6. Blue and orange curves represent vertical and horizontal position, respectively. Black dashed curves are the exact reference solutions. Constant extrapolation and coarse-grained interpolation are both unstable. Fine-grained interpolation results in stable simulation.



$$\begin{aligned}
 M_1 &= 100 \text{ kg} & B_1 &= 20 \text{ Ns/m} & L_1 &= 0.5 \text{ m} \\
 M_2 &= 100 \text{ kg} & B_2 &= 10 \text{ Ns/m} & L_2 &= 0.5 \text{ m} \\
 Z_{c,x} &= 1e5 \text{ Ns/m} & Z_{c,y} &= 1e5 \text{ Ns/m} & Z_{c,rot} &= 0 \text{ Nms}
 \end{aligned}$$

Fig. 2.5 A double pendulum is modelled as two arms connected by a TLM element. Impedance in the rotational dimension is set zero to allow free rotation

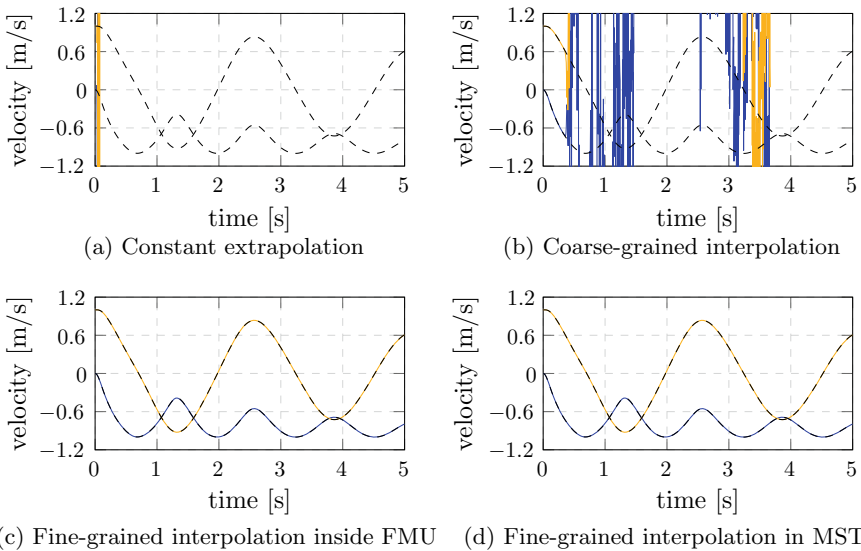
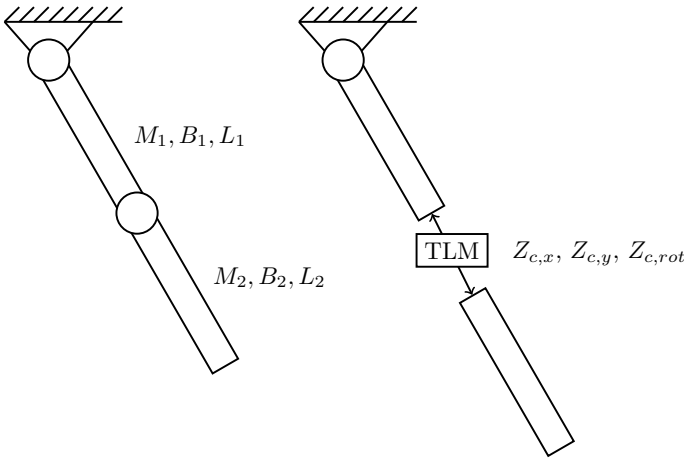


Fig. 2.6 Vertical and horizontal positions of the lower end of the double pendulum model. Dashed lines are the exact reference solutions

2.3.3 2D Single Pendulum

The final example composite model also consists of two pendulum arms connected through a TLM element. However, unlike the previous example model, the TLM element now has a rotational inertia of $Z_c = 1e4$. In this way the TLM element represents a fixed attachment rather than a joint. Hence, the two arms are attached to each other and will swing together like a single pendulum, see Fig. 2.7. In general, single pendulums are less demanding to simulate compared to double pendulums. This model is included to verify the methods against a rigid TLM connection, which is locked in all dimensions.

Simulation is initiated with the two arms pointing horizontally. Results with the four different methods are shown in Fig. 2.8. Blue and orange curves represent vertical and horizontal position, respectively. Black dashed curves are the exact reference solutions. In consistence with results from the previous composite models, extrapolation and coarse-grained interpolation are both unstable. Fine-grained interpolation results in stable simulation.



$$\begin{aligned}
 M_1 &= 100 \text{ kg} & B_1 &= 20 \text{ Ns/m} & L_1 &= 0.5 \text{ m} \\
 M_2 &= 100 \text{ kg} & B_2 &= 10 \text{ Ns/m} & L_2 &= 0.5 \text{ m} \\
 Z_{c,x} &= 1e5 \text{ Ns/m} & Z_{c,y} &= 1e5 \text{ Ns/m} & Z_{c,rot} &= 1e4 \text{ Nms}
 \end{aligned}$$

Fig. 2.7 A single pendulum is modelled as two arms connected by a stiff TLM element. Impedance is non-zero in all three dimensions

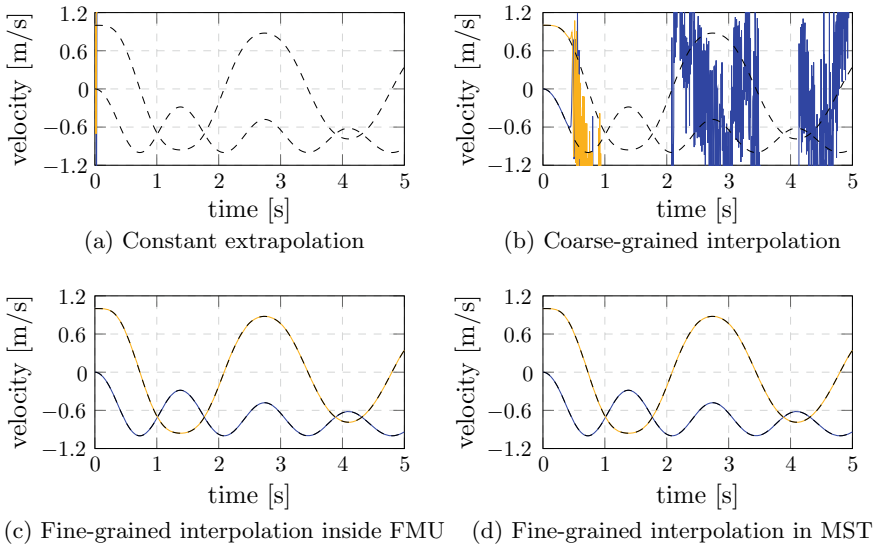


Fig. 2.8 Vertical and horizontal positions of the lower end of the single pendulum model. Dashed lines are the exact reference solutions

2.3.4 Thermodynamic Connection

The presented industrial application consists of two connected FMUs for co-simulation generated from one Modelica model. The two FMUs are connected via a TLM element with a characteristic impedance of $Z_c = 700000 \text{ sPa/m}^3$ and $\Delta t = 0.24 \text{ ms}$. These transmission line settings stem from a pipe with an approximate length of 0.1 m in which an ideal and incompressible gas flows. The physical quantities not accounted for in the TLM connection, in this case specific enthalpy and two phase water content, are passed in both directions directly between the two FMUs as delayed signal connections.

In Fig. 2.9, the left-hand side FMU instantiation acts as a source generating an oscillating mass flow to the second instantiation which acts as a sink. The FMU sink and source specific characteristics are specified via input parameters. The Modelica model in the application example is developed in the Modelica library Modelica

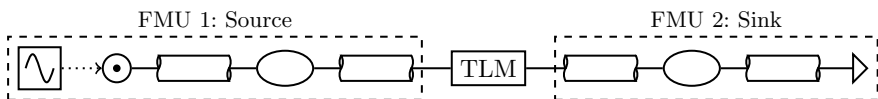


Fig. 2.9 Schematic description of industrial use-case. Two FMUs are connected via a TLM element. The FMUs originate from one Modelica model that is parametrized such that it can represent a source and a sink. The FMU is exported from the commercial Modelica-based modelling tool Dymola [6]

Fluid Light (MFL) [7]. MFL is particularly used for modelling of different industry-grade aircraft cooling and aircraft coolant distribution systems such as the aircraft vehicle systems simulator presented in [9]. The use-case in Fig. 2.9 is relevant as its FMU interface is principally equivalent to its more complex counterparts. MFL is an in-house library developed by Saab where information of mass flow, pressure, water content, and enthalpy are passed between resistive and capacitive components. The application example comprises several different differential and algebraic equations (DAEs). The resistive components, in this example the pipes, describe the mass flow as a function of component pressure drop by means of an algebraic equation. The capacitive components, in this case the volumes, express the system dynamics by means of a first order differential equation relating net mass flow to the pressure time derivative.

The composite model presented in Fig. 2.9 is simulated using constant extrapolation, coarse grained interpolation, and fine-grained interpolation inside the FMU. The latter two are achieved by means of modifying the Modelica model prior to FMU export. Two different adaptors are added to the Modelica model. The adaptor presented in Listing 2.5 receives information on wave variables from the MST. These values are equally distributed in time across the macro step. An interpolation table in the adaptor is populated at the start of each communication step; the pressure input to the original model is then available at all necessary internal times by means

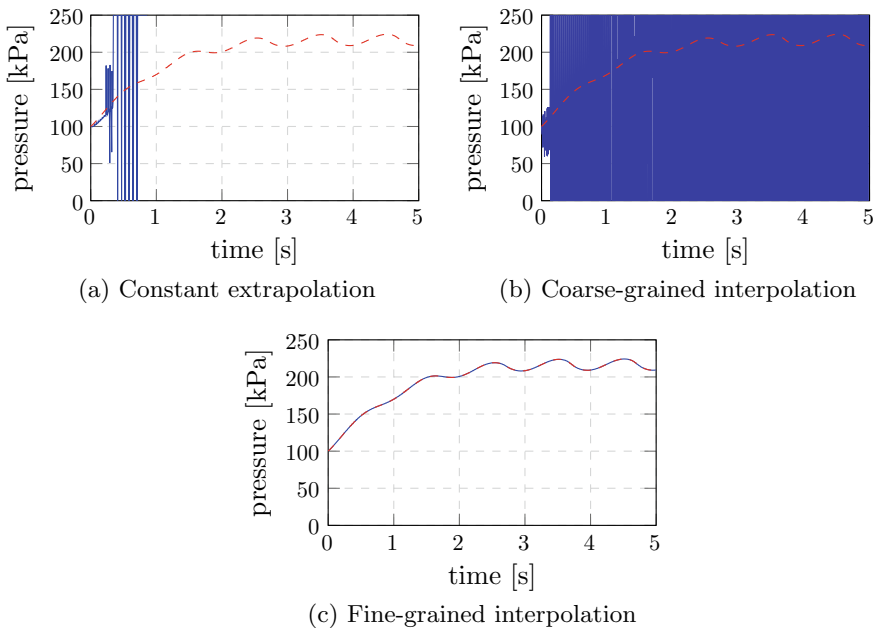


Fig. 2.10 Pressure in the TLM connection of the thermodynamic model. Dashed line is the Modelica simulation reference solution

of linear interpolation. The adaptor presented in Listing 2.4 receives information on wave variables and their corresponding time derivatives from the master at the beginning of each communication step. The composite model input pressure can then be estimated for any necessary internal times by means of the forward Euler method. Figure 2.10 clearly visualizes the advantage of having interpolated input information available as numerical stability is maintained when using fine-grained interpolation. Coarse-grained interpolation and constant extrapolation results in obvious stability issues. The simulation is terminated prematurely as a result of the severe stability issues resulting from constant extrapolation.

2.4 Conclusions

It is shown that fine-grained interpolation is required to achieve stable connections for all the presented composite models. Variables can either be interpolated locally inside the FMU, or be handled by the master simulation tool. The first method requires the FMU to be provided with the complete interpolation table. This leads to a large amount of data exchange, which may reduce simulation performance. Meanwhile, the second method would require a callback function from where the slaves can request interpolated data from the master simulation tool. Three possible improvements to the FMI standard that would facilitate asynchronous data exchange have been identified:

- Improved support for exchanging interpolation tables
- Support for callback functions
- Support for one-step execution mode

Fine-grained interpolation would be facilitated by the first two suggestions. Support for simple exchange of interpolation tables would enable interpolation of input variables inside an FMU. Callback functions on the other hand will provide the FMU with access to variables interpolated in the master simulation tool. An advantage with a callback function is that it is not limited to pure interpolation. In addition, it can include simple expressions, for example the TLM boundary equations. Having the interpolation table in the MST also significantly reduces the amount of data transfer compared to what is necessary if the tables are located inside the FMU. On the other hand, adding support for exchanging interpolation tables would constitute a far less comprehensive modification to the current standard. Finally, one-step execution mode will enable more densely populated interpolation tables and thereby improve accuracy in the interpolated variables.

References

1. Awais, M.U., Palensky, P., Mueller, W., Widl, E., Elsheikh, A.: Distributed hybrid simulation using the HLA and the functional mock-up interface. *Industrial Electronics Society, IECON*, pp. 7564–7569 (2013)
2. Bastian, J., Clauß, C., Wolf, S., Schneider, P.: Master for co-simulation using FMI. In: 8th International Modelica Conference, Dresden (2011)
3. Blochwitz, T., Otter, M., Arnold, M., Bausch, C., Clauß, C., Elmqvist, H., Junghanns, A., Mauss, J., Monteiro, M., Neidhold, T., Neumerkel, D., Olsson, H., Peetz, J.-V., Wolf, S.: The functional mockup interface for tool independent exchange of simulation models. In: 8th International Modelica Conference 2011, Como, Italy (2009)
4. Braun, R., Ericsson, L., Krus, P.: Full vehicle simulation of forwarder with semi active suspension using co-simulation. In: ASME/BATH 2015 Symposium on Fluid Power and Motion Control (2015)
5. Braun, R., Krus, P.: Tool-independent distributed simulations using transmission line elements and the Functional Mock-up Interface. In: 53rd SIMS conference on Simulation and Modelling, Reykjavik, Iceland (2013)
6. Brück, D., Elmqvist, H., Mattsson, S.E., Olsson, H.: Dymola for multi-engineering modeling and simulation. In: *Proceedings of Modelica*, vol. 2002 (2002)
7. Eek, Magnus, Gavel, Hampus, Ölvander, Johan: Definition and implementation of a method for uncertainty aggregation in component-based system simulation models. *J. Verif. Valid. Uncertain. Quantif.* **2**(1), 011006 (2017)
8. Elsheikh, A., Awais, M.U., Widl, E., Palensky, P.: Modelica-enabled rapid prototyping of cyber-physical energy systems via the functional mockup interface. In: 2013 Workshop on Modeling and Simulation of Cyber-Physical Energy Systems (MSCPES), pp. 1–6. IEEE (2013)
9. Hällqvist, R., Braun, R., Krus, P.: Early insights on FMI-based co-simulation of aircraft vehicle systems. In: The 15th Scandinavian International Conference on Fluid Power (2017)
10. Hindmarsh, A.C., Brown, P.N., Grant, K.E., Lee, S.L., Serban, R., Shumaker, D.E., Woodward, C.S.: SUNDIALS: Suite of nonlinear and differential/algebraic equation solvers. *ACM Trans. Math. Softw. (TOMS)* **31**(3), 363–396 (2005)
11. Krus, P.: Robust system modelling using bi-lateral delay lines. In: *Proceedings of the 2nd Conference on Modeling and Simulation for Safety and Security*. Linköping, Sweden (2005)
12. Neema, H., Gohl, J., Lattmann, Z., Sztipanovits, J., Karsai, G., Neema, S., Bapty, T., Batteh, J., Tummescheit, H., Sureshkumar, C.: Model-based integration platform for FMI co-simulation and heterogeneous simulations of cyber-physical systems. In: *Proceedings of the 10th International Modelica Conference*, vol. 096, pp. 235–245, 10–12 March 2014, Lund, Sweden. Linköping University Electronic Press (2014)
13. Schierz, T., Arnold, M., Clauß, C.: Co-simulation with communication step size control in an FMI compatible master algorithm. In: 9th International Modelica Conference, Munich, Germany, pp. 205–214 (2012)
14. Siemers, Alexander, Fritzon, Dag, Nakhimovski, Iakov: General meta-model based co-simulations applied to mechanical systems. *Simul. Model. Pract. Theory* **17**(4), 612–624 (2009)

Chapter 3

Local Extrapolation and Linear-Implicit Stabilization in a Parallel Coupling Scheme



Michael Burger and Stefan Steidel

Abstract We consider prediction strategies in a parallel coupling scheme for modular co-simulation: local extrapolation and a linear-implicit stabilization technique based on model information. That is, concerning local extrapolation, instead of using data points at the macro time points for generating the extrapolation polynomial (as it is done in the conventional global case), we use *local* data points only within the last macro time step. The linear-implicit stabilization technique predicts coupling quantities based on model information in terms of Jacobian matrices by performing a linear-implicit Euler step forward in time. We introduce and discuss these two prediction strategies and analyze their numerical properties, stability and accuracy, based on a simple test model.

3.1 Introduction

Most modern technical systems are of a complex and heterogeneous nature. In particular, that means, these systems may consist of multiple, dynamically interacting subsystems that originate from a variety of physical domains. For instance, a modern vehicle combines subsystems from mechanics (structure), electrics (e.g., headlights, air conditioning) and electronics (several controllers and driver assistance systems), possibly also from hydraulics, e.g., in case of commercial vehicles.

Accordingly, a numerical model of the full system has to cover all these subsystems. It is a common approach to model all subsystems independently using

M. Burger · S. Steidel (✉)

Department Dynamics, Loads and Environmental Data, Division Mathematics for Vehicle Engineering, Fraunhofer Institute for Industrial Mathematics ITWM, Fraunhofer-Platz 1, 67663 Kaiserslautern, Germany

e-mail: stefan.steidel@itwm.fraunhofer.de

M. Burger

e-mail: michael.burger@itwm.fraunhofer.de

© Springer Nature Switzerland AG 2019

B. Schweizer (ed.), *IUTAM Symposium on Solver-Coupling and Co-Simulation*, IUTAM Bookseries 35, https://doi.org/10.1007/978-3-030-14883-6_3

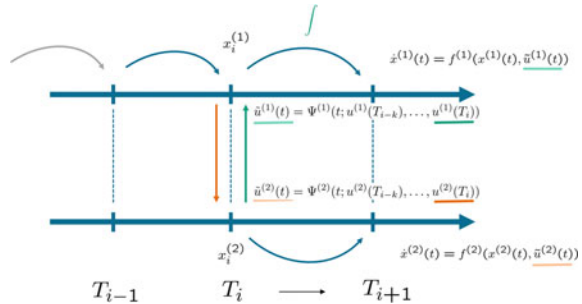
modelling techniques, environments and languages that are suited best for the modelling process within the corresponding physical domain. All the resulting *submodels* are usually given in form of differential equations. However, they may differ substantially, e.g., in their formulations, in their time-constants and time-scales as well as in their complexity. If all model equations are available, at least in terms of right-hand side calls, one could combine all those to an entire (monolithic) system of model equations and solve the latter with a single integration scheme. In most cases, this strategy is, however, not advisable due to the mentioned heterogeneity in the numerical properties and characteristics. Thus, it is preferable to use specifically tailored numerical solvers for each submodel, in order to realize an efficient simulation. Additionally, in realistic applications, it might be the case that the submodels are built-up within commercial software tools, which only provide slender interfaces to a combination of model and solver, rather than the possibility for right-hand side calls.

The key property of coupled simulation strategies—also referred to as co-simulation, modular or distributed simulation—is not only to model the subsystem independently, but also to solve them *separately* on consecutive time windows. Herein, the time integration of each subsystem model within one time-window can be done with different stepsizes adapted to the subsystem (multirate approach), or even with problem- and domain-specific solvers (multimethod approach). During the integration process of one subsystem, the needed coupling quantities, i.e., inputs from other subsystems, are approximated, usually by predictions based on known results and information from the past. Co-simulation strategies are under investigation for many years, for an overview, we refer to the survey papers [10, 13] as well as to [5, 7] and the references therein for a brief introduction of several coupling strategies and techniques. More detailed numerical analyses can be found, e.g., in [3–6, 8, 11, 12].

In this contribution, we address the prediction task. We focus on a parallel coupling scheme, a so-called *Jacobi scheme*, [5, 8]. That is, the involved subsystems are solved independently and in parallel; they use extrapolated, predicted input (coupling) data, which are stored and exchanged on a fixed macro time grid $\mathbb{G} := \{T_0, \dots, T_N\}$ with $t_0 = T_0 < T_1 < \dots < T_N = t_f$, cf. Fig. 3.1.

In particular and without loss of generality, we consider here the case of two coupled subsystems, a generalization to multiple subsystems is achieved straight-

Fig. 3.1 Parallel coupling scheme (Jacobi scheme): Two subsystems are executed and solved in parallel, information is exchanged only at macro time points T_i



forwardly. We assume that both systems are mathematically described each by an ordinary differential equation (ODE) with, in general, nonlinear output quantities,

$$\dot{x}^{(1)} = f^{(1)}(x^{(1)}, u^{(1)}) \quad (3.1)$$

$$y^{(1)} = g^{(1)}(x^{(1)}, u^{(1)}) \quad (3.2)$$

$$\dot{x}^{(2)} = f^{(2)}(x^{(2)}, u^{(2)}) \quad (3.3)$$

$$y^{(2)} = g^{(2)}(x^{(2)}, u^{(2)}) \quad (3.4)$$

$$u = Ly. \quad (3.5)$$

Again without loss of generality, we may further assume a linear coupling with a non-singular matrix $L \in \mathbb{R}^{n_u \times n_y}$ with

$$u := \begin{bmatrix} u^{(1)} \\ u^{(2)} \end{bmatrix} \in \mathbb{R}^{n_u}, \quad y := \begin{bmatrix} y^{(1)} \\ y^{(2)} \end{bmatrix} \in \mathbb{R}^{n_y} \quad (3.6)$$

$n_u = n_{u^{(1)}} + n_{u^{(2)}} \in \mathbb{N}$ and $n_y = n_{y^{(1)}} + n_{y^{(2)}} \in \mathbb{N}$. Together with the coupling condition as stated in (3.5), Eqs. (3.1)–(3.5) form a (nonlinear) differential-algebraic equation (DAE), which, in principle, could be solved as monolithic system. However, as already indicated, here we focus on a parallel execution of both subsystems together with a parallel coupling scheme.

The remaining part of this paper is organized as follows. In Sect. 3.2, we introduce two prediction strategies, that can be used for extrapolating coupling quantities. Section 3.3 contains a numerical analysis, in terms of stability and accuracy, of those two prediction approaches, on the basis of a simple test model. In the last Sect. 3.4, we summarize the presented approaches and results.

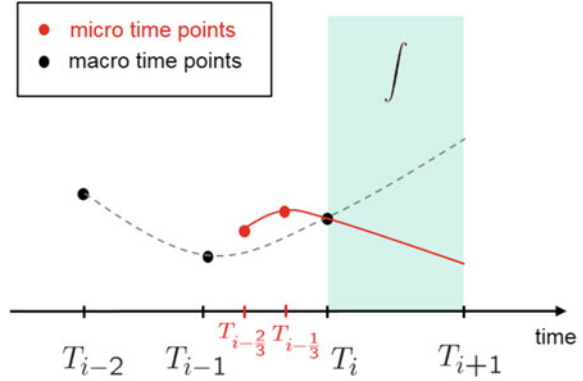
3.2 Prediction Strategies

Within this section we present two prediction strategies that are suited for extrapolating coupling data within a parallel coupling scheme as introduced in the Sect. 3.1.

3.2.1 Local Extrapolation

We consider a local extrapolation approach in a parallel coupling scheme. That is, instead of using data points at the macro time points T_i for generating the extrapolation polynomial (as it is done in the global case), we use data points only within the last macro time step. More precisely, for the extrapolation used in the time step $T_i \rightarrow T_{i+1}$, we only use (several) data points of the coupling quantities within the interval $[T_{i-1}, T_i]$, as illustrated in Fig. 3.2.

Fig. 3.2 Local quadratic extrapolation (red)



We define micro time points $T_{i-\delta} := T_i - \delta H$ and let $n \in \mathbb{N}$ be the polynomial degree of the considered extrapolation. Then we consider data points (T_{i-j}, Y_{i-j}) for $j = 0, \dots, n$ in the global case, and data points (T_{i-p_j}, Y_{i-p_j}) for $j = 0, \dots, n$ with $p_j \in [0, 1]$ in the local case. Hence, we obtain extrapolation polynomials for each subsystem

$$\mu_i(s) = \begin{bmatrix} \mu_{1i}(s) \\ \mu_{2i}(s) \end{bmatrix} = \sum_{j=0}^n v_{ij} s^j, \quad v_{ij} = \begin{bmatrix} v_{1ij} \\ v_{2ij} \end{bmatrix} \in \mathbb{R}^{n_u}, \quad s \in [T_i, T_{i+1}] \quad (3.7)$$

satisfying $u_{i-j} = \mu_i(T_{i-j}) = Ly_{i-j}$ and $u_{i-p_j} = \mu_i(T_{i-p_j}) = Ly_{i-p_j}$, respectively. Note that we choose the same extrapolation order n for each subsystem.

Comparing both approaches, for the local extrapolation, we need output data *within* the macro time steps as well, whereas in the global approach only data *at* the macro time points has to be exchanged. However, this should not be a severe restriction concerning practicability and application situations.

3.2.2 Linear-Implicit Stabilization

We analyze the linear-implicit stabilization approach for co-simulation. Again, we concentrate on the parallel Jacobi scheme. The crucial idea of that approach is to make predictions based on model information of the other involved co-simulation partners. To be more precise, model information shall be shared in terms of Jacobian matrices, cf. [2, 14]. In these days, a new interface standard has been developed, the so-called "Functional Mock-Up Interface (FMI) for Model-Exchange and Co-Simulation", cf. [1]. This interface is supported by several commercial CAE software tools and finds more and more interest in industry for application projects. It also provides the possibility to exchange directional derivatives, from which the Jacobian matrices can be constructed.

We assume that the Jacobian matrices $\partial f^{(i)}/\partial x^{(i)}$, $\partial f^{(i)}/\partial u^{(i)}$, $\partial g^{(i)}/\partial x^{(i)}$, $\partial g^{(i)}/\partial u^{(i)}$, $i = 1, 2$, of the involved co-simulation partners are available and can be exchanged at the macro time points. Setting

$$x := \begin{bmatrix} x^{(1)} \\ x^{(2)} \end{bmatrix}, \quad f := \begin{bmatrix} f^{(1)} \\ f^{(2)} \end{bmatrix}, \quad g := \begin{bmatrix} g^{(1)} \\ g^{(2)} \end{bmatrix}, \quad (3.8)$$

the basic step of this stabilization approach is to perform a linear-implicit time step of length H (i.e. macro stepsize), in order to obtain a prediction for the coupling quantities. In particular, an *implicit Euler-step* of length H would take the form

$$x_{i+1} = x_i + Hf(x_{i+1}, u_{i+1}), \quad (3.9)$$

or, equivalently,

$$x_{i+1} = x_i + \Delta x_{i+1}, \quad \Delta x_{i+1} := Hf(x_{i+1}, u_{i+1}). \quad (3.10)$$

We come to the linear-implicit step by setting

$$f(x_{i+1}, u_{i+1}) \approx f(x_i, u_i) + \frac{\partial f}{\partial x} \Delta x_{i+1} + \frac{\partial f}{\partial u} \Delta u_{i+1}. \quad (3.11)$$

The coupling condition leads to

$$\Delta u_{i+1} = L\Delta y_{i+1}. \quad (3.12)$$

And the linearized (implicit) output relation takes the form

$$g(x_{i+1}, u_{i+1}) \approx \Delta y_{i+1} = C\Delta x_{i+1} + D\Delta u_{i+1} = C\Delta x_{i+1} + DL\Delta y_{i+1}, \quad (3.13)$$

assuming without loss of generality $g(x_i, u_i) = 0$ and defining

$$C := \left. \frac{\partial g}{\partial x} \right|_{(x_i, u_i)}, \quad D := \left. \frac{\partial g}{\partial u} \right|_{(x_i, u_i)}. \quad (3.14)$$

Thus,

$$\Delta y_{i+1} = (\mathbb{1} - DL)^{-1} C \Delta x_{i+1}, \quad (3.15)$$

$$\Delta u_{i+1} = L(\mathbb{1} - DL)^{-1} C \Delta x_{i+1}, \quad (3.16)$$

and, consequently,

$$\begin{aligned} \Delta x_{i+1} &= H \left(f(x_i, u_i) + A\Delta x_{i+1} + B\Delta u_{i+1} \right) \\ &= H \left(f(x_i, u_i) + A\Delta x_{i+1} + BL(\mathbb{1} - DL)^{-1} C \Delta x_{i+1} \right), \end{aligned} \quad (3.17)$$

with

$$A := \frac{\partial f}{\partial x} \Big|_{(x_i, u_i)}, \quad B := \frac{\partial f}{\partial u} \Big|_{(x_i, u_i)}. \quad (3.18)$$

Whence, we obtain

$$\underbrace{(\mathbb{1} - HA - HBL(\mathbb{1} - DL)^{-1}C)}_{=: \Omega_E} \Delta x_{i+1} = Hf(x_i, u_i), \quad (3.19)$$

and, last, not least,

$$\Delta y_{i+1} = (\mathbb{1} - DL)^{-1}C \Delta x_{i+1} = H \underbrace{(\mathbb{1} - DL)^{-1}C \Omega_E^{-1}}_{=: \psi_E} f(x_i, u_i). \quad (3.20)$$

This leads to a predicted output

$$y_{i+1}^{\text{pred}} := y_i + \Delta y_{i+1} = y_i + H \psi_E f(x_i, u_i), \quad (3.21)$$

and in the macro time step $T_i \rightarrow T_{i+1}$, we can interpolate inbetween y_i and y_{i+1}^{pred} .

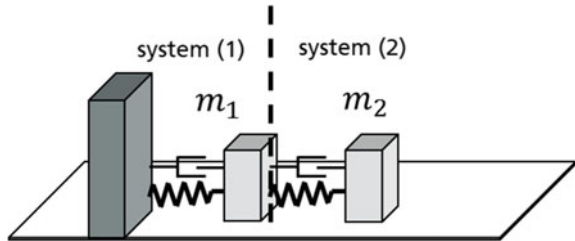
3.3 Numerical Study

We analyze the stability and accuracy of the introduced prediction approaches in Sect. 3.2 using a linear two-mass spring-damper (2-MSD) test problem with masses m_1, m_2 , stiffness constants k_1, k_2 , damping constants d_1, d_2 and force-displacement coupling in a parallel co-simulation scheme, cf. Fig. 3.3 and [8, 9].

In particular, we consider the following monolithic system

$$\dot{x} = A_{\text{Mon}} x, \quad x = \begin{pmatrix} x^{(1)} \\ \dot{x}^{(1)} \\ x^{(2)} \\ \dot{x}^{(2)} \end{pmatrix}, \quad A_{\text{Mon}} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ -\frac{k_1+k_2}{m_1} & -\frac{d_1+d_2}{m_1} & \frac{k_2}{m_1} & \frac{d_2}{m_1} \\ 0 & 0 & 0 & 1 \\ \frac{k_2}{m_2} & \frac{d_2}{m_2} & -\frac{k_2}{m_2} & -\frac{d_2}{m_2} \end{pmatrix}. \quad (3.22)$$

Fig. 3.3 Two-mass spring-damper test model



Decomposing the 2-MSD test problem results in the strongly coupled system of the form

$$\begin{aligned} \dot{x} &= Ax + Bu \\ y &= Cx + Du \\ u &= Ly \end{aligned} \quad (3.23)$$

with states $x = (x^{(1)}, \dot{x}^{(1)}, x^{(2)}, \dot{x}^{(2)})^T \in \mathbb{R}^4$, inputs $u = (u^{(1)}, u_1^{(2)}, u_2^{(2)})^T \in \mathbb{R}^3$, outputs $y = (y_1^{(1)}, y_2^{(1)}, y_2^{(2)})^T \in \mathbb{R}^3$ and time-invariant system matrices

$$\begin{aligned} A &= \left(\begin{array}{cc|cc} 0 & 1 & & \\ \hline -\frac{k_1}{m_1} & -\frac{d_1}{m_1} & & \\ & & 0 & 1 \\ & & \hline & & -\frac{k_2}{m_2} & -\frac{d_2}{m_2} \end{array} \right) \in \mathbb{R}^{4 \times 4}, & B &= \left(\begin{array}{c|cc} 0 & & \\ \hline \frac{1}{m_1} & & \\ & 0 & 0 \\ & \hline & \frac{k_2}{m_2} & \frac{d_2}{m_2} \end{array} \right) \in \mathbb{R}^{4 \times 3}, \\ C &= \left(\begin{array}{cc|cc} 1 & 0 & & \\ \hline 0 & 1 & & \\ & & & \\ & & k_2 & d_2 \end{array} \right) \in \mathbb{R}^{3 \times 4}, & D &= \left(\begin{array}{c|cc} 0 & & \\ \hline 0 & & \\ & & \\ & & \hline & -k_2 & -d_2 \end{array} \right) \in \mathbb{R}^{3 \times 3}, \\ & & L &= \left(\begin{array}{c|cc} & & 1 \\ \hline 1 & 0 & \\ \hline 0 & 1 & \end{array} \right) \in \mathbb{R}^{3 \times 3}. \end{aligned} \quad (3.24)$$

In this considered example we fix $m_1 = 10 \text{ kg}$, $k_1 = 10^6 \text{ N/m}$, $d_1 = 1 \text{ Ns/m}$, $m_2 = 20 \text{ kg}$ and vary $k_2 \in [1, 10^6] \text{ N/m}$, $d_2 \in [1, 10^6] \text{ Ns/m}$, $H \in [10^{-5}, 10^{-3}] \text{ s}$. We choose a simulation time of one second, i.e. $T \in [0, 1] \text{ s}$, and consider a reference solution x_{ref} by solving the monolithic test system (3.22) via

$$x_{\text{ref}}(T_{i+1}) = e^{A_{\text{Mon}} H_{\text{ref}}} x_{\text{ref}}(T_i), \quad (3.25)$$

with $H_{\text{ref}} = 10^{-6} \text{ s}$ and initial value $x_{\text{ref}}(0) = (0.1, 0, 0, 0)^T$.

3.3.1 Global and Local Extrapolation

Now, considering the strongly coupled system (3.23) with data points (T_{i-j}, y_{i-j}) and (T_{i-p_j}, y_{i-p_j}) with $p_j \in [0, 1]$ for $j = 0, \dots, n$, respectively, and applying the well-known *Lagrange polynomials* ℓ_j^n for n data points satisfying

$$\ell_j^n(s) = \prod_{k=0, k \neq j}^n \frac{s - T_{i-k}}{T_{i-j} - T_{i-k}}, \quad (3.26)$$

we end up with extrapolation polynomials of the following form:

$$\mu_i(s) = \sum_{j=0}^n u_{i-j} \ell_j^n(s) = \sum_{j=0}^n L y_{i-j} \ell_j^n(s) = L \sum_{j=0}^n y_{i-j} \ell_j^n(s), \quad s \in [T_i, T_{i+1}]. \quad (3.27)$$

Thus, the solution of the time-integration $T_i \rightarrow T_{i+1} = T_i + H$ is given by

$$\begin{aligned} x_{i+1} = x(H) &= e^{AH} x_i + \int_0^H e^{A(H-s)} B \mu_i(s) ds \\ &= e^{AH} x_i + \int_0^H e^{A(H-s)} B \left(L \sum_{j=0}^n y_{i-j} \ell_j^n(s) \right) ds \\ &= e^{AH} x_i + \sum_{j=0}^n y_{i-j} \underbrace{\left(\int_0^H e^{A(H-s)} B L \ell_j^n(s) ds \right)}_{=: \alpha_j^n = \alpha_j^n(H) \in \mathbb{R}^{n_x \times n_y}} \\ &= e^{AH} x_i + \sum_{j=0}^n \alpha_j^n y_{i-j} \end{aligned} \quad (3.28)$$

where α_j^n can be pre-computed as long as n is chosen, since we only compute the Lagrange polynomials ℓ_j^n once in advance for the time points $\{0, -H, \dots, -nH\}$. Consequently, the discrete propagation of Eq. (3.23) for the outputs together with $u_{i+1}^{\text{pred}} = \mu_i(H)$ and Eq. (3.27) leads to the following form:

$$\begin{aligned} y_{i+1} &= C x_{i+1} + D u_{i+1}^{\text{pred}} \\ &= C \left(e^{AH} x_i + \sum_{j=0}^n \alpha_j^n y_{i-j} \right) + D \left(L \sum_{j=0}^n y_{i-j} \ell_j^n(H) \right) \\ &= C e^{AH} x_i + \sum_{j=0}^n \underbrace{\left(C \alpha_j^n + D L \ell_j^n(H) \right)}_{=: \beta_j^n = \beta_j^n(H) \in \mathbb{R}^{n_y \times n_y}} y_{i-j} \\ &= C e^{AH} x_i + \sum_{j=0}^n \beta_j^n y_{i-j} \end{aligned} \quad (3.29)$$

The above equations together yield the following matrix-vector representation

$$\begin{pmatrix} x_{i+1} \\ y_{i+1} \end{pmatrix} = \begin{pmatrix} e^{AH} \\ C e^{AH} \end{pmatrix} \cdot x_i + \sum_{j=0}^n \begin{pmatrix} \alpha_j^n \\ \beta_j^n \end{pmatrix} \cdot y_{i-j}. \quad (3.30)$$

Setting

$$G_n(H) := \begin{pmatrix} e^{AH} & \alpha_0^n & 0 & \alpha_1^n & \dots & 0 & \alpha_n^n \\ Ce^{AH} & \beta_0^n & 0 & \beta_1^n & \dots & 0 & \beta_n^n \end{pmatrix} \quad (3.31)$$

we generally obtain

$$\begin{pmatrix} x_{i+k} \\ y_{i+k} \end{pmatrix} = G_n(kH) \cdot (x_i, y_i, x_{i-1}, y_{i-1}, \dots, x_{i-n}, y_{i-n})^T. \quad (3.32)$$

and particularly the following iteration matrices for constant, linear and quadratic global and local polynomial extrapolation, i.e. for $n = 0, 1, 2$.

1. Constant extrapolation ($n = 0$):

$$\begin{pmatrix} x_{i+1} \\ y_{i+1} \end{pmatrix} = G_0(H) \cdot \begin{pmatrix} x_i \\ y_i \end{pmatrix} = \Gamma_{\text{global}(0)}(H) \cdot \begin{pmatrix} x_i \\ y_i \end{pmatrix}. \quad (3.33)$$

2. Linear global polynomial extrapolation ($n = 1$):

$$\begin{pmatrix} x_{i+1} \\ y_{i+1} \\ x_i \\ y_i \end{pmatrix} = \underbrace{\begin{pmatrix} G_1(H) \\ G_1(0) \end{pmatrix}}_{=: \Gamma_{\text{global}(1)}(H)} \cdot \begin{pmatrix} x_i \\ y_i \\ x_{i-1} \\ y_{i-1} \end{pmatrix}. \quad (3.34)$$

3. Linear local polynomial extrapolation ($n = 1$):

$$\begin{pmatrix} x_{i+1} \\ y_{i+1} \\ x_{i-p+1} \\ y_{i-p+1} \end{pmatrix} = \underbrace{\begin{pmatrix} G_1(H) \\ G_1((1-p)H) \end{pmatrix}}_{=: \Gamma_{\text{local}(1,p)}(H)} \cdot \begin{pmatrix} x_i \\ y_i \\ x_{i-p} \\ y_{i-p} \end{pmatrix}. \quad (3.35)$$

4. Quadratic global polynomial extrapolation ($n = 2$):

$$\begin{pmatrix} x_{i+1} \\ y_{i+1} \\ x_i \\ y_i \\ x_{i-1} \\ y_{i-1} \end{pmatrix} = \underbrace{\begin{pmatrix} G_2(H) \\ G_2(0) \\ G_2(-H) \end{pmatrix}}_{=: \Gamma_{\text{global}(2)}(H)} \cdot \begin{pmatrix} x_i \\ y_i \\ x_{i-1} \\ y_{i-1} \\ x_{i-2} \\ y_{i-2} \end{pmatrix}. \quad (3.36)$$

5. Quadratic local polynomial extrapolation ($n = 2$):

$$\begin{pmatrix} x_{i+1} \\ y_{i+1} \\ x_{i-p_1+1} \\ y_{i-p_1+1} \\ x_{i-p_2+1} \\ y_{i-p_2+1} \end{pmatrix} = \underbrace{\begin{pmatrix} G_2(H) \\ G_2((1-p_1)H) \\ G_2((1-p_2)H) \end{pmatrix}}_{=: \Gamma_{\text{local}(2,p_1,p_2)}(H)} \cdot \begin{pmatrix} x_i \\ y_i \\ x_{i-p_1} \\ y_{i-p_1} \\ x_{i-p_2} \\ y_{i-p_2} \end{pmatrix}. \quad (3.37)$$

To guarantee a stable co-simulation, it is necessary that $\rho(\Gamma_{\text{global}(n)}(H)) \leq 1$ and $\rho(\Gamma_{\text{local}(n,p)}(H)) \leq 1$, respectively, with ρ denoting the spectral radius.

3.3.2 Linear-Implicit Stabilization with Implicit Euler-Step

Applying the linear-implicit stabilization approach as described in Sect. 3.2.2, particularly Eq. (3.20), to our linear test-problem (3.23), we obtain

$$y_{i+1}^{\text{pred}} = y_i + H \psi_E [Ax_i + BLy_i] = (1 + H \psi_E BL) y_i + (H \psi_E A) x_i. \quad (3.38)$$

If we choose linear interpolation between y_i and y_{i+1}^{pred} , we arrive at the formula

$$\begin{aligned} y(s) &= \left(1 - \frac{s}{H}\right) y_i + \frac{s}{H} y_{i+1}^{\text{pred}} \\ &= \left(1 - \frac{s}{H}\right) y_i + \frac{s}{H} \left((1 + H \psi_E BL) y_i + (H \psi_E A) x_i \right) \\ &= (\psi_E As) x_i + (1 + \psi_E BLs) y_i \end{aligned} \quad (3.39)$$

for $s \in [0, H]$. Substituting Eq. (3.39) into (3.28) leads to

$$x_{i+1} = e^{AH} x_i + \int_0^H e^{A(H-s)} BLy(s) ds = \alpha_E x_i + \beta_E y_i \quad (3.40)$$

with

$$\alpha_E := \alpha_E(H) = e^{AH} + \int_0^H e^{A(H-s)} BL \psi_E As ds \in \mathbb{R}^{n_x \times n_x}, \quad (3.41)$$

$$\beta_E := \beta_E(H) = \int_0^H e^{A(H-s)} BL(\mathbb{1} + \psi_E BLs) ds \in \mathbb{R}^{n_x \times n_y}. \quad (3.42)$$

Together with $y_{i+1}^{\text{pred}} = y(H)$ following Eq. (3.39) and

$$\begin{aligned} y_{i+1} &= Cx_{i+1} + DLy_{i+1}^{\text{pred}} \\ &= C(\alpha_E x_i + \beta_E y_i) + DL((\psi_E AH)x_i + (1 + \psi_E BLH)y_i) \\ &= (C \alpha_E + DL \psi_E AH)x_i + (C \beta_E + DL(1 + \psi_E BLH))y_i \end{aligned} \quad (3.43)$$

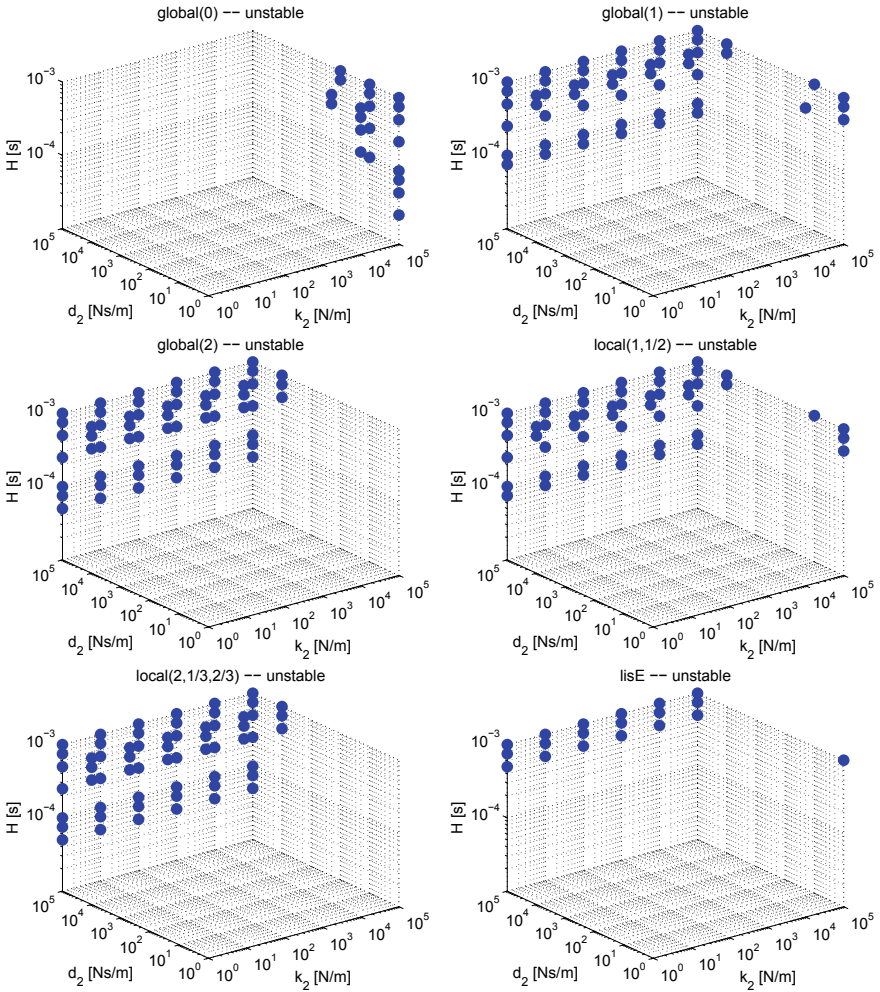


Fig. 3.4 Unstable regions ($\rho > 1$) for the 2-MSD test problem with $m_1 = 10$ kg, $k_1 = 10^6$ N/m, $d_1 = 1$ Ns/m and $m_2 = 20$ kg

we can establish an iteration formula of the form

$$\begin{pmatrix} x_{i+1} \\ y_{i+1} \end{pmatrix} = \Gamma_E(H) \cdot \begin{pmatrix} x_i \\ y_i \end{pmatrix} \tag{3.44}$$

with

$$\Gamma_E(H) := \begin{pmatrix} \alpha_E & \beta_E \\ C \alpha_E + DL \psi_E AH & C \beta_E + DL(1 + \psi_E BLH) \end{pmatrix}. \tag{3.45}$$

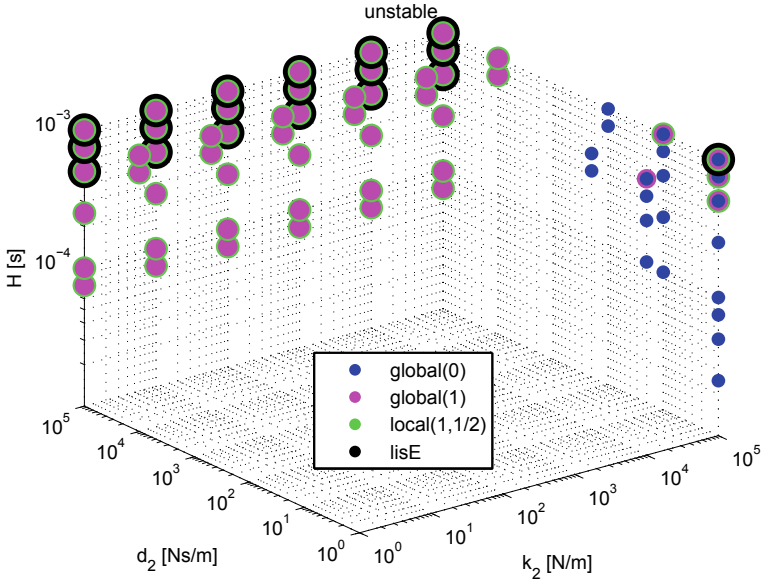


Fig. 3.5 Unstable regions ($\rho > 1$) for the 2-MSD test problem with $m_1 = 10$ kg, $k_1 = 10^6$ N/m, $d_1 = 1$ Ns/m and $m_2 = 20$ kg

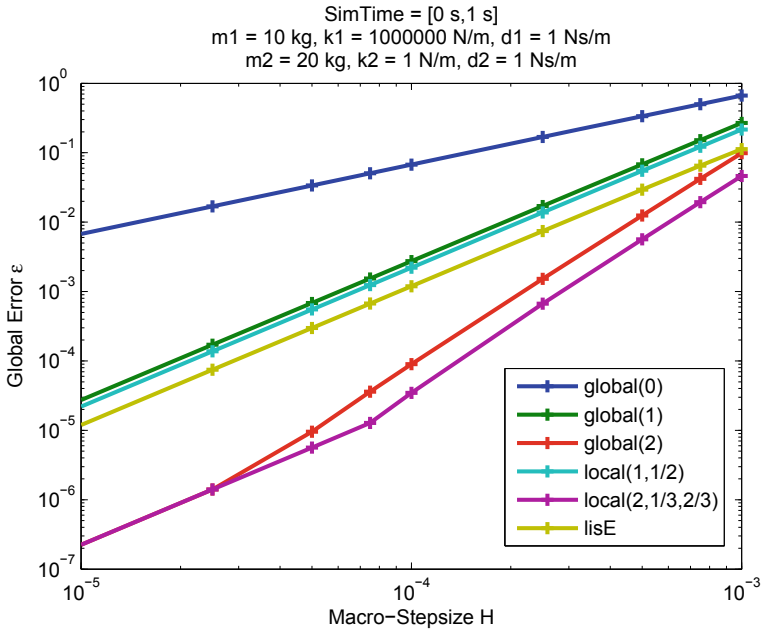


Fig. 3.6 Global errors for the 2-MSD test problem with $m_1 = 10$ kg, $k_1 = 10^6$ N/m, $d_1 = 1$ Ns/m, $m_2 = 20$ kg, $k_2 = 1$ N/m, $d_2 = 1$ Ns/m

Again, to guarantee a stable co-simulation, it is necessary that $\rho(\Gamma_E(H)) \leq 1$ with ρ denoting the spectral radius. Unstability regions, i.e., points in the parameter configuration space, for which it holds that $\rho > 1$ are shown in Figs. 3.4 and 3.5.

In order to address the accuracy of the introduced prediction approaches, we define the *global error* ε as follows:

$$\varepsilon = \max_{0 \leq T_i \leq 1} \left(\sum_{j=1}^2 \|x_i^{(j)} - x_{\text{ref}}^{(j)}(T_i)\| + \|\dot{x}_i^{(j)} - \dot{x}_{\text{ref}}^{(j)}(T_i)\| \right). \quad (3.46)$$

The global error behaviour with respect to the 2-MSD test problem for the exemplary parameters $m_1 = 10 \text{ kg}$, $k_1 = 10^6 \text{ N/m}$, $d_1 = 1 \text{ Ns/m}$, $m_2 = 20 \text{ kg}$, $k_2 = 1 \text{ N/m}$, $d_2 = 1 \text{ Ns/m}$ is depicted in Fig. 3.6.

3.4 Conclusion

In this contribution, two prediction strategies have been discussed and analyzed, the local extrapolation and the linear-implicit Euler step, where the latter extrapolates coupling data based on model information in terms of Jacobian matrices. Both approaches are discussed with a special focus on the parallel Jacobi coupling scheme, in which all involved subsystems are solved independently in parallel, i.e., each subsystem needs extrapolated coupling data in each macro step. Both extrapolation methods may, however, also be used in other coupling schemes, such as schemes of Gauss-Seidel-type or in dynamic iteration schemes. In Sect. 3.3, both strategies are analyzed concerning their stability and accuracy based on a simple test model, which is often used for the investigation of co-simulation approaches.

References

1. <https://www.fmi-standard.org/>
2. Arnold, M.: Multi-rate time integration for large scale multibody system models. In: IUTAM Symposium on Multiscale Problems in Multibody System Contacts, held in Stuttgart, Germany (2006)
3. Arnold, M.: Stability of sequential modular time integration methods for coupled multibody system models. J. Comput. Nonlinear Dyn. **5**, 031003 (2010)
4. Arnold, M., Clauß, C., Schierz, T.: Error analysis and error estimates for co-simulation in FMI for model exchange and co-simulation v2.0. Arch. Mech. Eng. **LX**, 75–94 (2013)
5. Arnold, M., Günther, M.: Preconditioned dynamic iteration for coupled differential-algebraic systems. BIT Numer. Math. **41**(1), 001–025 (2001)
6. Bartel, A., Brunk, M., Schüps, S.: On the convergence rate of dynamic iteration for coupled problems with multiple subsystems. J. Comput. Appl. Math. **262**, 14–24 (2014). Selected Papers from NUMDIFF-13

7. Burger, M., Gerdt, M.: A survey on numerical methods for the simulation of initial value problems with DAEs. In: Ilchmann, A., Reis, T. (eds.) *Surveys in Differential-Algebraic Equations IV. Differential-Algebraic Equations Forum*, pp. 221–300. Springer International Publishing, Cham (2017)
8. Busch, M.: *Zur Effizienten Kopplung von Simulationsprogrammen*. Kassel University Press (2012)
9. Busch, M., Schweizer, B.: Numerical stability and accuracy of different co-simulation techniques: Analytical investigations based on a 2-dof test model. In: *The 1st Joint International Conference on Multibody System Dynamics*, Lappeenranta, Finland, 25–27 May 2010
10. Jackson, K.R.: A survey of parallel numerical methods for initial value problems for ordinary differential equations. *IEEE Trans. Mag.* **27**(5), 3792–3797 (1991)
11. Kübler, R., Schiehlen, W.: Two methods of simulator coupling. *Math. Comput. Model. Dyn. Syst.* **6**(2), 93–113 (2000)
12. Schierz, T., Arnold, M.: Stabilized overlapping modular time integration of coupled differential-algebraic equations. *Appl. Numer. Math.* **62**(10), 1491–1502 (2012). Selected Papers from NUMDIFF-12
13. Veitl, A., Gordon, T., van de Sand, A., Howell, M., Valášek, M., Vaculín, O., Steinbauer, P.: Methodologies for coupling simulation models and codes in mechatronic system analysis and design. In: *Proceedings of the 16th IAVSD-Symposium on Dynamics of Vehicles on Roads and Tracks*. Pretoria, volume 33 of *Supplement to Vehicle System Dynamics*, pp. 231–243. Swets & Zeitlinger (1999)
14. Viel, A.: Implementing stabilized co-simulation of strongly coupled systems using the functional mock-up interface 2.0. In: *Proceedings of the 10th International Modelica Conference*, Lund, Sweden (2014)

Chapter 4

Performance Improvement of Explicit Co-simulation Methods Through Continuous Extrapolation



Martin Busch

Abstract In order to couple several simulation models, the corresponding software tools can be interconnected by means of a co-simulation. The inputs and outputs of the models depend on each other and have to be updated during the time integration process of the numerical solvers. Since the tools can only communicate at discrete macro-time points, the model inputs are mostly approximated, e.g., by using polynomial interpolation and extrapolation techniques. As a drawback of classical extrapolation methods, discontinuities occur at the macro-time points. This can slow down the solvers and reduces the efficiency of the co-simulation. The current paper considers continuous approximation techniques of C^0 , C^1 and C^2 type which are capable to overcome the discontinuity issues. The approaches are analyzed regarding numerical stability, global error and performance. To show the benefit of the continuity, the methods are implemented in a master-slave co-simulation and a comparison with the classical discontinuous approach is done. The C^2 -continuous approach mostly outperforms the methods of lower continuity. The C^0 -continuous method fails due to a limitation of the error order. With a here-presented enhancement the order drop of the C^0 -continuous method can be avoided.

4.1 Introduction

Co-simulation, also named “tool coupling” or “simulator coupling”, is frequently used to simulate multidisciplinary problems that comprise subsystems from various physical domains, see e.g. Refs. [1, 2, 8, 11, 13, 16, 19, 33, 35, 37]. The subsystems are modeled and simulated in different suitable simulation tools which are connected by a coupling interface, such as FMI¹ or TISC.² The tools communicate with each other only sporadically, i.e., the coupling data of the subsystems are interchanged

¹www.modelisar.org.

²www.tlk-thermo.com.

M. Busch (✉)
Wiesenstraße 28, 91469 Hagenbüchach, Germany
e-mail: busch@research-multibodydynamics.de

only at discrete macro-time points. Therefore, the numerical solvers of the tools can operate largely independent—using problem-adopted integration methods and optimized step sizes—which can be very advantageous for the calculation effort, see e.g. the multirate approaches in Refs. [3, 20, 22, 24, 36] or the parallelization techniques in Refs. [7, 12, 17, 32, 34].

Several numerical approaches are available for implementing a co-simulation, such as explicit approaches, see Refs. [6, 9, 10, 20, 27], full-implicit approaches, see Refs. [5, 25], or semi-implicit approaches, see Refs. [3, 11, 23, 30, 31, 36]. Implicit approaches are based on a repetition of the macro steps which makes them very stable and robust, see Ref. [9]. However, the step repetition requires the subsystem solvers to reinitialize the time integration at former macro-time points. The most commercial simulation tools do not support this reinitialization feature which is a main reason why implicit co-simulation approaches are still rarely implemented in state-of-the-art coupling interfaces. In contrast, explicit co-simulation approaches calculate the macro steps only forward in time (“explicitly”) and can hence be implemented very easily in commercial simulation tools. Thus, the present paper focuses on explicit methods.

The use of explicit co-simulation methods bears some disadvantages. Since the simulation tools can only interchange the coupling data at the discrete macro-time points, the explicit coupling interface acts like a filter and the numerical solution can be subject to time delays, artificial noise and energy loss, see Ref. [6]. Applying energy-preserving co-simulation methods, these issues may be circumvented, see Refs. [6, 18, 21].

Further, explicit coupling approaches introduce numerical instabilities so that the numerical solution can rise exponentially in time, see Refs. [4, 9, 28]. These instabilities force the user to work with very small macro-step sizes to meet a sufficient accuracy. As mentioned above, applying implicit methods may strongly improve the numerical stability and the efficiency of the co-simulation. Alternatively, adaptive co-simulation techniques may be applied to achieve the best accuracy and performance, either by controlling the macro-step size, see Refs. [9, 26, 29, 36], or by switching the numerical methods during the simulation, see Refs. [15, 21].

Since explicit methods can only step forward in time, extrapolation techniques have to be applied to predict the coupling variables in the macro steps.³ Classical extrapolation techniques lead to discontinuities in the solution which arise after each macro step and disturb the numerical time integration in the subsystems. If the solvers reduce their step size or reject the steps at the discontinuities, the calculation effort can strongly increase.

In Ref. [14], a C^0 -continuous extrapolation method with the name *extrapolated interpolation* was proposed to avoid the discontinuity issue. In Ref. [10], the method was enhanced to a C^1 -continuous extrapolation method. Further, it was shown that the original C^0 -continuous method has a reduced error order which leads to reduced

³Using a sequential Gauss-Seidel method for the co-simulation, one subsystem obtains extrapolated and the other subsystem obtains interpolated coupling variables while both subsystems obtain extrapolated coupling variables if a parallel Jacobi method is applied.

accuracy compared with the C^1 -continuous approach. While the C^0 -continuous approach was already implemented and tested in Ref. [8], an implementation of the C^1 -continuous approach within a co-simulation interface has not been done, so far.

In the current paper, the continuous methods are enhanced once again. The original C^0 -continuous method is improved to preserve the full error order and the C^1 -continuous approach is enhanced to C^2 -continuity. All continuous methods are implemented in a master-slave co-simulation interface and the performance is investigated and compared to the classical discontinuous approach. Further, all approaches are compared regarding stability and error. A linear test model is used for the investigations.

Regarding common techniques for system decomposition, the here-presented co-simulation methods are mainly tailored to couplings based on applied forces, such as force-displacement or displacement-displacement couplings. Constraint couplings, which are based on reaction forces and algebraic equations, can principally also be handled by the approaches, but the application might not be useful since constraint couplings are in general not well-conditioned for the use of explicit co-simulation methods, because the zero-stability of the methods may not be ensured.⁴ Thus, constraint couplings are not considered in the present paper.

In Sect. 4.2, the test model is briefly discussed and the coupling structure is explained. In Sect. 4.3, the continuous methods are introduced and enhanced. The numerical stability of the methods is calculated in Sect. 4.4. The subsystems are solved with an analytical time integration method. In Sect. 4.5, the methods are implemented in a master-slave co-simulation and the global error as well as the performance are compared. The subsystems are calculated with different numerical solvers of varying accuracy.

4.2 Linear Test Model and System Decomposition

In order to apply a co-simulation to a multidisciplinary system, the system must be decomposed into several subsystems by defining appropriate coupling variables. For sake of simplicity, in the present paper we consider a linear test system of two degrees of freedom, i.e., a 2-mass oscillator, see Fig. 4.1. The parameters of the oscillator are the masses m_1 and m_2 , the stiffness c_1 and c_2 , the damping coefficients d_1 and d_2 as well as the stiffness c_k and the damping coefficient d_k of the coupling spring between the two masses. For the decomposition of the system we use a force-displacement coupling where the second subsystem calculates the applied coupling force f_k and

⁴For constraint couplings, the zero-stability of the co-simulation method depends on the structure of the submodels, e.g., on the ratio of masses. Several stabilization techniques exist to generate zero-stable explicit methods for constraint couplings, see e.g. the overlapping technique in Ref. [5]. However, these methods are mostly based on an adaption of the model equations in the subsystems which is hardly realizable in commercial simulation tools. In contrast, for applied-force couplings the zero-stability of explicit co-simulation methods is always guaranteed as long as the subsystem solvers are zero-stable and the applied forces do not depend on accelerations, see Refs. [9, 25].

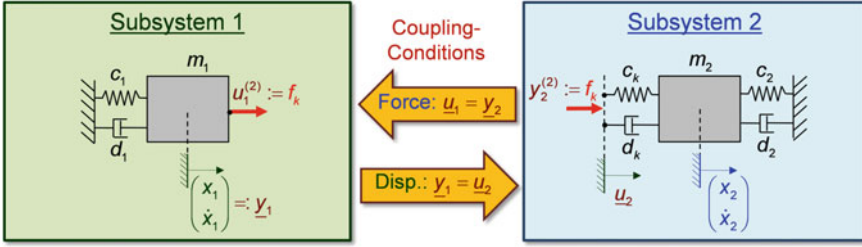


Fig. 4.1 Modular modeling of 2-DOF oscillator using a force-displacement coupling

the first subsystem yields the position x_1 and the velocity \dot{x}_1 of the first mass. The system decomposition results in the following linear equations in state-space form

$$\dot{z}_1 = \underline{A}_1 \cdot z_1 + \underline{B}_1 \cdot u_1, \quad \dot{z}_2 = \underline{A}_2 \cdot z_2 + \underline{B}_2 \cdot u_2, \quad (4.1a)$$

$$y_1 = \underline{C}_1 \cdot z_1 + \underline{D}_1 \cdot u_1, \quad y_2 = \underline{C}_2 \cdot z_2 + \underline{D}_2 \cdot u_2, \quad (4.1b)$$

with the subsystem states $z_1 = \begin{pmatrix} x_1 \\ \dot{x}_1 \end{pmatrix} \in \mathbb{R}^2$ and $z_2 = \begin{pmatrix} x_2 \\ \dot{x}_2 \end{pmatrix} \in \mathbb{R}^2$ as well as the input and output vectors $u_1, u_2, y_1, y_2 \in \mathbb{R}^2$ of the subsystems. The coupling conditions

$$u_1 = y_2 \quad , \quad u_2 = y_1 \quad (4.1c)$$

accomplish that the output of each subsystem is connected with the input of the other subsystem. The subsystem matrices $\underline{A}_i, \underline{B}_i, \underline{C}_i,$ and \underline{D}_i ($i = 1, 2$) of the force-displacement coupling can be found in Ref. [9], for instance, and shall be given here only in abstract form.

With the force-displacement coupling in Fig. 4.1, the output of the first subsystem does not explicitly depend on the input (no *direct feed-through*). Thus, the feed-through matrix \underline{D}_1 is zero which guarantees that all numerical coupling methods, considered in this paper, will be zero-stable if zero-stable solvers are used in the subsystems, see Ref. [9, 25].

4.3 Approximation Methods for Co-simulation

Applying a co-simulation method to equation system (4.1), the state equations (4.1a) are calculated by appropriate subsystem solvers. The input vectors u_1, u_2 have to be updated during the time integration. For this purpose, the time is discretized into macro-time points T_i ($i = 1, 2, 3, \dots$) where the subsystems exchange their coupling data. Hence, the coupling conditions (4.1c) are only satisfied at T_i . Between the

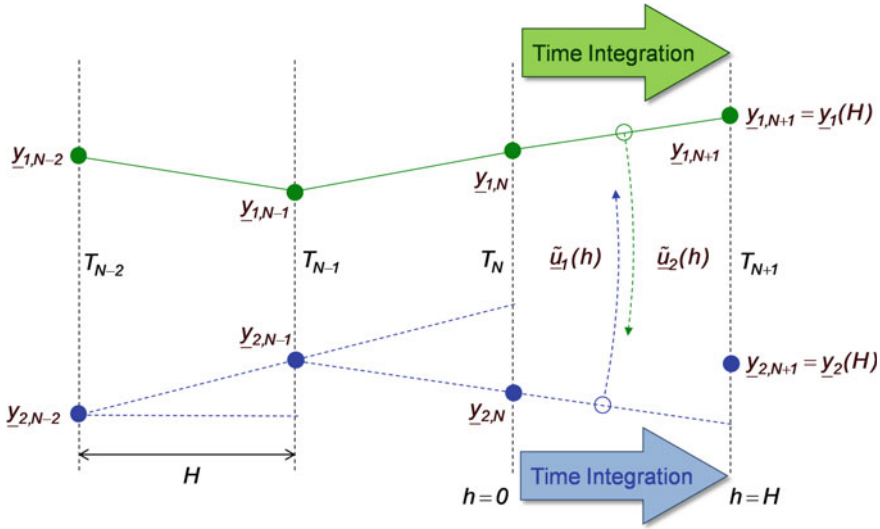


Fig. 4.2 Sequential Gauss-Seidel scheme using a macro step $T_N \rightarrow T_{N+1}$ with constant macro-step size H and a classical extrapolation polynomial (blue-dotted curves) for generating inputs for subsystem 1

macro-time points, the unknown inputs are approximated (*weak coupling*). In the following, different approximation approaches are described.

4.3.1 Classical Discontinuous Approach

Let's consider the macro step $T_N \rightarrow T_{N+1}$ with constant macro-step size $H = T_{N+1} - T_N$, see Fig.4.2. Using a sequential Gauss-Seidel scheme for the co-simulation, the unknown inputs \underline{u}_1 of the first subsystem are approximated with an extrapolation polynomial (blue-dotted curve) while the inputs \underline{u}_2 of the second subsystem are approximated with an interpolation polynomial (green-solid curve). The polynomials are calculated from the discrete output vectors \underline{y}_1 and \underline{y}_2 at the macro-time points. Mathematically, a classical extrapolation for the input of subsystem 1 can be expressed by a Lagrange-polynomial function Φ_1 of degree k which depends on the output data $\underline{y}_{2,N-k}, \dots, \underline{y}_{2,N}$ of subsystem 2 at the $k + 1$ previous macro-time points

$$\begin{aligned} \tilde{\underline{u}}_1^{(k)}(h) &:= \Phi_1 \left([-k \cdot H, \dots, -H, 0], \left[\underline{y}_{2,N-k}, \dots, \underline{y}_{2,N-1}, \underline{y}_{2,N} \right]; h \right) \\ &= \sum_{i=0}^k \underline{y}_{2,N-i} \cdot L_{k-i}^{(k)}(h) \quad \text{(Classical polynomial extrapolation in subsystem 1).} \end{aligned} \tag{4.2}$$

In the second subsystem, the output data $\underline{y}_{1,N-j+1}, \dots, \underline{y}_{1,N+1}$ from subsystem 1 can be used for the input approximation, including the updated value $\underline{y}_{1,N+1}$ at the next macro-time point T_{N+1} (or at $h = H$ in terms of step sizes). The interpolated input for subsystem 2 can be expressed by a Lagrange-polynomial function $\underline{\Phi}_2$ of degree j

$$\begin{aligned} \tilde{u}_2^{(j)}(h) &= \underline{\Phi}_2 \left([-(j-1) \cdot H, \dots, -H, 0, H], [\underline{y}_{1,N+1-j}, \dots, \underline{y}_{1,N-1}, \underline{y}_{1,N}, \underline{y}_{1,N+1}]; h \right) \\ &= \sum_{i=0}^j \underline{y}_{1,N+1-i} \cdot L_{j-i}^{(j)}(h-H) \quad (\text{Classical polynomial interpolation in subsystem 2}). \end{aligned} \quad (4.3)$$

The Lagrange basis polynomials of degree d are defined as

$$L_m^{(d)}(h) := \begin{cases} 0 & \text{if } m < 0 \\ 0 & \text{if } m > d \\ \prod_{\substack{n=0 \\ n \neq m}}^d \frac{h - h_n}{h_m - h_n} & \text{otherwise} \end{cases}, \quad (4.4)$$

with the equidistant macro-step sizes $h_n := -(d-n) \cdot H, n = 0, \dots, d$.

As a drawback of classical co-simulation approaches, the extrapolation polynomials are discontinuous at the end of each macro step, see the gaps between the consecutive blue-dotted curves in Fig. 4.2. Consequently, the subsystem solvers may strongly reduce their integration-step size at the macro-time points. If the discontinuities are too large, i.e., if the macro-step size is chosen too large, the time integration can even abort.

4.3.2 Original C^0 -Continuous EXTRIPOL Approach

To overcome the discontinuity issue, a so-called *extrapolated interpolation* (EXTRIPOL) approach was introduced by Rauh and Dronka in Ref.[14], see Fig. 4.3. In each macro step, a classical polynomial extrapolation is carried out to obtain coupling data at T_N and T_{N+1} (blue circles). These extrapolated data are smoothed with a linear interpolation polynomial to obtain $\tilde{u}_1(h)$ (red-dashed curves). Hence, the approximated input of subsystem 1 can be expressed by the following system of equations

$$\begin{aligned} \underline{\Phi}_{1,N} &:= \underline{\Phi}_1 \left([-(k+1) \cdot H, \dots, -2H, -H], [\underline{y}_{2,N-(k+1)}, \dots, \underline{y}_{2,N-2}, \underline{y}_{2,N-1}]; h = 0 \right) \\ &= \sum_{i=0}^k \underline{y}_{2,N-1-i} \cdot L_{k-i}^{(k)}(h-H) \Big|_{h=0} \quad (\text{Classical polynomial extrapolation at } T_N), \\ \underline{\Phi}_{1,N+1} &:= \underline{\Phi}_1 \left([-k \cdot H, \dots, -H, 0], [\underline{y}_{2,N-k}, \dots, \underline{y}_{2,N-1}, \underline{y}_{2,N}]; h = H \right) \\ &= \sum_{i=0}^k \underline{y}_{2,N-i} \cdot L_{k-i}^{(k)}(h) \Big|_{h=H} \quad (\text{Classical polynomial extrapolation at } T_{N+1}), \\ \tilde{u}_1^{(k)}(h) &:= \underline{\Phi}_{1,N} \cdot L_0^{(1)}(h-H) + \underline{\Phi}_{1,N+1} \cdot L_1^{(1)}(h-H) \quad (C^0\text{-approximation in subsystem 1}). \end{aligned} \quad (4.5)$$

Note that this procedure yields a C^0 -continuous input for subsystem 1 over the macro steps.

Using the original C^0 -continuous method, the degree of the smoothing polynomial remains linear even if the degree of the underlying polynomial is increased, see Fig. 4.3b. In Ref. [10], the numerical error of the method was investigated and it was shown that this limitation reduces the order of the co-simulation error significantly. Consequently, very small macro-step sizes have to be used with the C^0 -continuous method to achieve the same accuracy as the classical discontinuous method. To avoid the order reduction, the degree of the smoothing polynomial must be increased which can be accomplished with the following approaches.

4.3.3 C^1 and C^2 -Continuous EXTRIPOL Approach

A first approach was presented in Ref. [10] which uses a Hermite polynomial for the smoothing, see Fig. 4.4. The Hermite polynomial (red-dashed curve) is computed from the extrapolated points (blue circles) and the first derivative (blue arrows) of the underlying polynomials. Hence, a cubic smoothing polynomial is obtained which is not only C^0 -continuous but even C^1 -continuous over the macro steps, i.e., the first derivative of the red-dashed curve is continuous. The corresponding equations for the input approximation in subsystem 1 are

$$\begin{aligned}
 \dot{\Phi}_{1,N} &:= \sum_{i=0}^k y_{2,N-1-i} \cdot \left. \frac{d}{dt} L_{k-i}^{(k)}(h-H) \right|_{h=0} && \text{(1st time derivative of } \Phi_{1,N}), \\
 \dot{\Phi}_{1,N+1} &:= \sum_{i=0}^k y_{2,N-i} \cdot \left. \frac{d}{dt} L_{k-i}^{(k)}(h) \right|_{h=H} && \text{(1st time derivative of } \Phi_{1,N+1}), \\
 \tilde{u}_1^{(k)}(h) &:= \Phi_{1,N} \cdot H_{0,0}^{(1)}(h-H) + \Phi_{1,N+1} \cdot H_{1,0}^{(1)}(h-H) \\
 &\quad + \dot{\Phi}_{1,N} \cdot H_{0,1}^{(1)}(h-H) + \dot{\Phi}_{1,N+1} \cdot H_{1,1}^{(1)}(h-H) && (C^1\text{-approx. in subsystem 1}),
 \end{aligned} \tag{4.6}$$

with $\Phi_{1,N}$ and $\Phi_{1,N+1}$ taken from Eq. (4.5). The Hermite-basis polynomials for the equidistant sampling points are defined by the equations

$$\begin{aligned}
 H_{m,0}^{(d)}(h) &:= [L_m^{(d)}(h)]^2 \cdot \left(1 - 2 \cdot \left. \frac{d}{dh} L_m^{(d)}(h) \right|_{h=h_m} \cdot (h-h_m) \right), \\
 H_{m,1}^{(d)}(h) &:= [L_m^{(d)}(h)]^2 \cdot (h-h_m)
 \end{aligned} \tag{4.7}$$

which yield a polynomial of degree $2 \cdot d + 1$, i.e., the smoothing polynomial in Eq. (4.6) is cubic since $d = 1$.

Regarding the numerical effort of the co-simulation, a higher continuity can be of additional advantage since the subsystem solvers are less disturbed. The approach (4.6) can easily be enhanced to C^2 or higher continuity by taking into account the curvature or higher derivatives of the underlying polynomials. In multibody applications, which are based on positions, velocities and accelerations of the bodies,

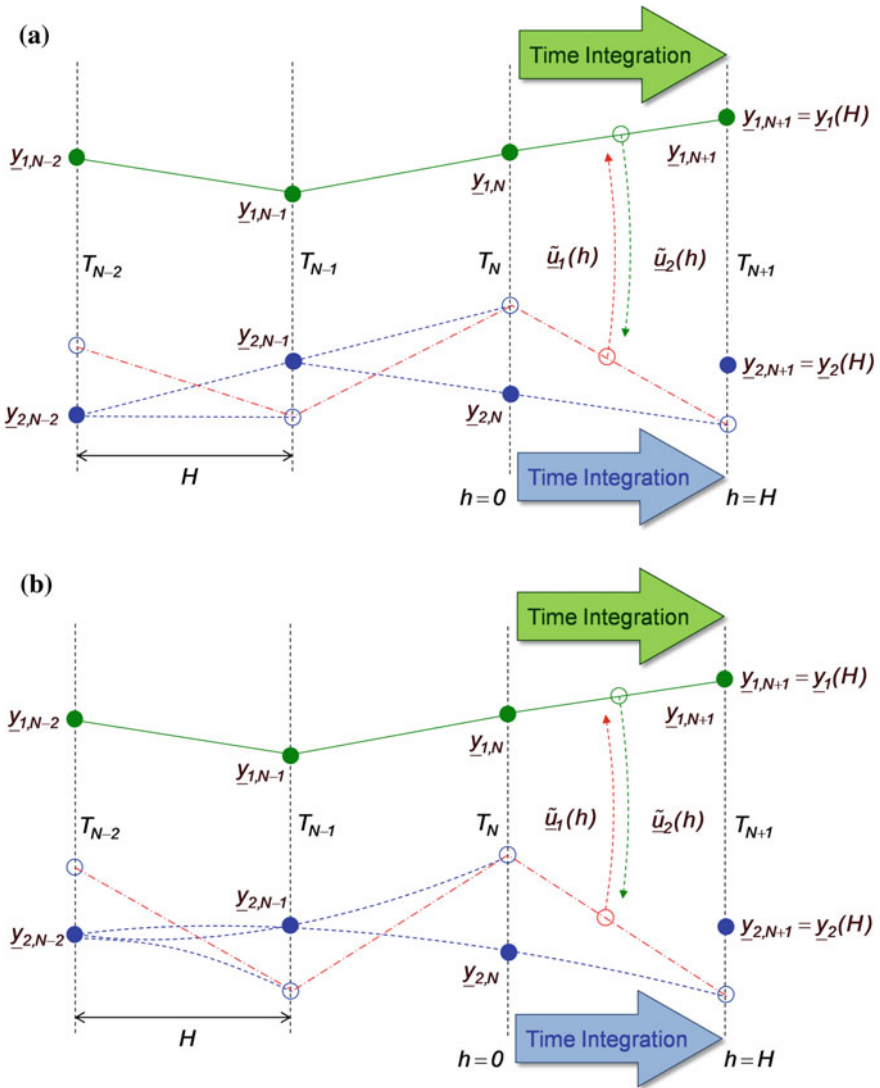


Fig. 4.3 Sequential Gauss-Seidel scheme using a macro step $T_N \rightarrow T_{N+1}$ with constant macro-step size H and a C^0 -continuous EXTRIPOL method (red, dashed curves) for generating inputs for subsystem 1. In subfigure (a) a linear polynomial and in subfigure (b) a quadratic polynomial is used for the underlying extrapolation (blue-dotted curves)

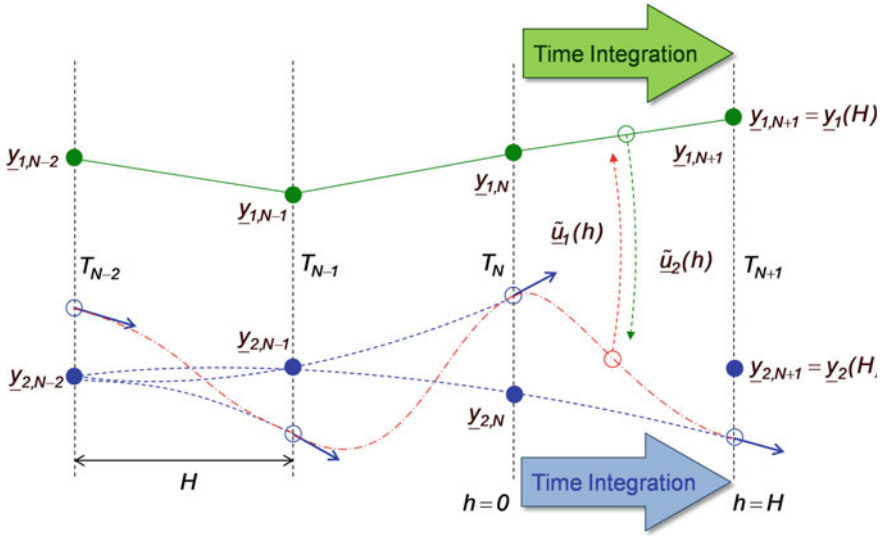


Fig. 4.4 Sequential Gauss-Seidel scheme using a macro step $T_N \rightarrow T_{N+1}$ with constant macro-step size H and a C^1 -continuous EXTRIPOL method (red-dashed curve) for generating inputs for subsystem 1

C^2 -continuity should mostly be sufficient for the solvers to not notice any discontinuity in the subsystem input, at all. The corresponding equations for a C^2 -continuous method are

$$\begin{aligned}
 \ddot{\underline{\Phi}}_{1,N} &:= \sum_{i=0}^k y_{2,N-1-i} \cdot \frac{d^2}{dt^2} L_{k-i}^{(k)}(h-H) \Big|_{h=0} && \text{(2nd time derivative of } \underline{\Phi}_{1,N}), \\
 \ddot{\underline{\Phi}}_{1,N+1} &:= \sum_{i=0}^k y_{2,N-i} \cdot \frac{d^2}{dt^2} L_{k-i}^{(k)}(h) \Big|_{h=H} && \text{(2nd time derivative of } \underline{\Phi}_{1,N+1}), \\
 \tilde{u}_1^{(k)}(h) &:= \underline{\Phi}_{1,N} \cdot H_{0,0}^{(1)}(h-H) + \underline{\Phi}_{1,N+1} \cdot H_{1,0}^{(1)}(h-H) \\
 &\quad + \underline{\dot{\Phi}}_{1,N} \cdot H_{0,1}^{(1)}(h-H) + \underline{\dot{\Phi}}_{1,N+1} \cdot H_{1,1}^{(1)}(h-H) \\
 &\quad + \underline{\ddot{\Phi}}_{1,N} \cdot H_{0,2}^{(1)}(h-H) + \underline{\ddot{\Phi}}_{1,N+1} \cdot H_{1,2}^{(1)}(h-H) \quad (C^2\text{-approx. in subsystem 1}),
 \end{aligned} \tag{4.8}$$

with $\underline{\Phi}_{1,N}, \underline{\Phi}_{1,N+1}$ taken from Eq. (4.5) and $\underline{\dot{\Phi}}_{1,N}, \underline{\dot{\Phi}}_{1,N+1}$ taken from Eq. (4.6). The Hermite-basis polynomials for higher continuity than C^1 can be defined implicitly by the interpolation condition

$$\frac{d^v}{dh^v} H_{m,\mu}^{(d)}(h) \Big|_{h=h_n} = \begin{cases} 1 & \text{if } v = \mu \wedge n = m \\ 0 & \text{otherwise} \end{cases} \tag{4.9}$$

which yields a polynomial of degree $(v+1) \cdot (d+1) - 1$. Hence, in Eq. (4.8), where $v = 2$ and $d = 1$, a quintic smoothing polynomial is obtained.

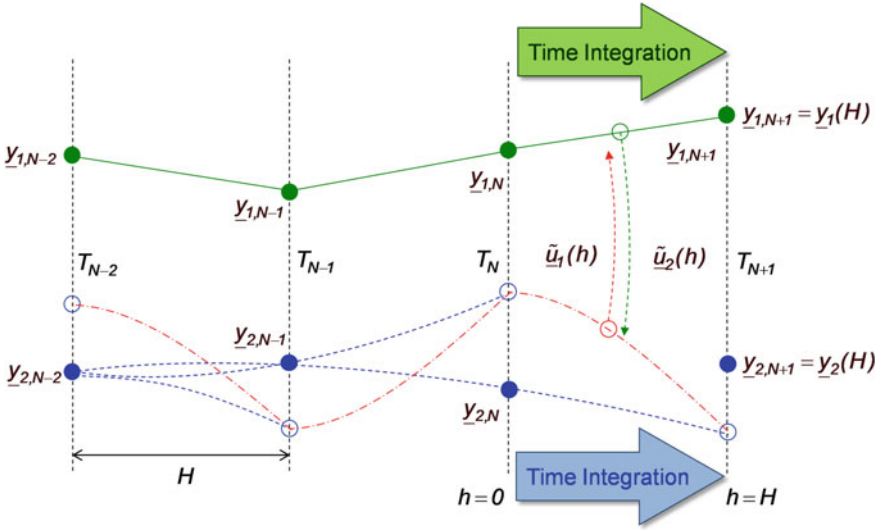


Fig. 4.5 Sequential Gauss-Seidel scheme using a macro step $T_N \rightarrow T_{N+1}$ with constant macro-step size H and the enhanced C^0 -continuous EXTRIPOL method (red-dashed curves) for generating inputs for subsystem 1. The red smoothing polynomials are only shown in the intervals they are used for and not for the complete range of sampling points

4.3.4 Enhanced C^0 -Continuous EXTRIPOL Approach

As a second approach to increase the degree of the smoothing polynomial, a classic piecewise Lagrange polynomial can be used, see the red-dashed curves in Fig. 4.5. To generate a polynomial of degree k , $k + 1$ extrapolated sampling points (blue circles) are taken into account which leads to the equations

$$\begin{aligned}
 \Phi_{1,N-k+1} &:= \Phi_1 \left([-2k \cdot H, \dots, -k \cdot H], [y_{2,N-2k}, \dots, y_{2,N-k}]; h = -(k-1) \cdot H \right) \\
 &= \sum_{i=0}^k y_{2,N-k-i} \cdot L_{2k-i}^{(k)}(h - -k \cdot H) \Big|_{h = -(k-1) \cdot H} \quad \text{(Classical polynomial extrapolation at } T_{N-k+1}\text{)}, \\
 &\vdots \\
 \Phi_{1,N} &:= \Phi_1 \left([-(k+1) \cdot H, \dots, -2H, -H], [y_{2,N-(k+1)}, \dots, y_{2,N-2}, y_{2,N-1}]; h = 0 \right) \\
 &= \sum_{i=0}^k y_{2,N-1-i} \cdot L_{k-i}^{(k)}(h - -H) \Big|_{h = 0} \quad \text{(Classical polyn. extrapolation at } T_N\text{)}, \\
 \Phi_{1,N+1} &:= \Phi_1 \left([-k \cdot H, \dots, -H, 0], [y_{2,N-k}, \dots, y_{2,N-1}, y_{2,N}]; h = H \right) \\
 &= \sum_{i=0}^k y_{2,N-i} \cdot L_{k-i}^{(k)}(h) \Big|_{h = H} \quad \text{(Classical polyn. extrapolation at } T_{N+1}\text{)}, \\
 \tilde{u}_1^{(k)}(h) &:= \Phi_1 \left([-(k-1) \cdot H, \dots, -H, 0, H], [\Phi_{1,N-k+1}, \dots, \Phi_{1,N}, \Phi_{1,N+1}]; h \right) \\
 &= \sum_{i=0}^k \Phi_{1,N-i+1} \cdot L_{k-i}^{(k)}(h - H) \quad \text{(C}^0\text{-approximation in subsystem 1)}.
 \end{aligned} \tag{4.10}$$

As a disadvantage of the method, many macro-steps have to be computed in order to build a smoothing polynomial of high degree. For instance, a cubic polynomial (degree $k = 3$) requires four extrapolated sampling points which are each calculated by cubic underlying extrapolation polynomials, hence, $2k + 1 = 7$ macro steps have to be carried out at first to apply the cubic method. Despite the high amount of necessary sampling points, the smoothing polynomial is still only C^0 -continuous over the macro steps which is a further disadvantage compared to the methods of Sect. 4.3.3.

4.3.5 Approximation Error of the Methods

In Fig. 4.6, the different methods are applied to a simple sine-function (black curve) to compare the approximation results. Cubic polynomials are used for the underlying extrapolation (blue curves). In plot (a) the classical discontinuous approach is shown where Lagrange polynomials are used for the extrapolation. In plot (b) the original and in plot (c) the enhanced C^0 -continuous EXTRIPOL method are shown (red

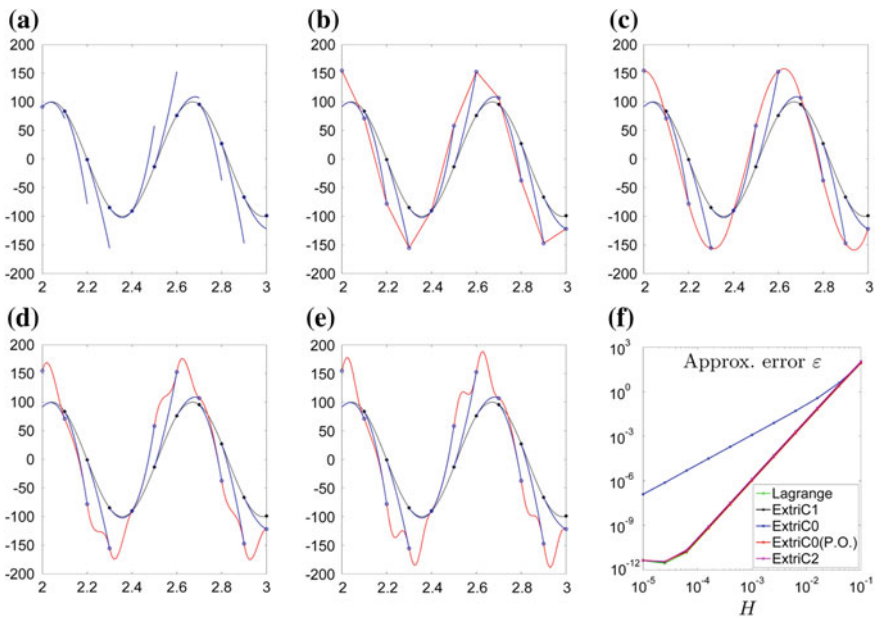


Fig. 4.6 Approximation methods applied to a sine function (black curve): the classical discontinuous approach in (a), the original C^0 -continuous EXTRIPOL method in (b), the enhanced C^0 -continuous EXTRIPOL method in (c), the C^1 -continuous EXTRIPOL method in (d) and the C^2 -continuous EXTRIPOL method in (e). In plot (f) the approximation error of the methods is logarithmically plotted over the step size

curves). Plots (d) and (e) show the C^1 and the C^2 -continuous EXTRIPOL approach where Hermite polynomials are used for the smoothing. It can be observed that the C^1 and C^2 -continuous methods tend to additional oscillations in the solution. However, these oscillations decay if the step size is reduced and the approximation polynomials converge to the sine function. In plot (f), the approximation error ε of the methods is shown over the step size H . The enhanced C^0 -continuous EXTRIPOL method (“*ExtriC0(P.O.)*”) as well as the C^1 and the C^2 -continuous EXTRIPOL methods (“*ExtriC1*” and “*ExtriC2*”) generate the same error order as the classical discontinuous method (“*Lagrange*”) which has order 4.⁵ Only for the original C^0 -continuous EXTRIPOL method, the error order is limited to an order of 2 due to the use of linear smoothing polynomials.

4.4 Numerical Stability of the Co-simulation

In the following, it is investigated how the approximation methods from Sect. 4.3 affect the numerical stability of the co-simulation. For the classical discontinuous approach this analysis was carried out in Ref. [9], for the C^1 -continuous EXTRIPOL method a corresponding analysis can be found in Ref. [10]. In the same manner, the numerical stability for the enhanced C^0 -continuous and the C^2 -continuous EXTRIPOL method shall be analyzed. The calculation procedure is only coarsely described here, details can be found in the mentioned references.

The approximation methods from Sect. 4.3 are applied to the linear test model from Sect. 4.2. Hence, in the subsystem Eqs. (4.1) the inputs are replaced by the approximated inputs

$$\dot{z}_1 = \underline{A}_1 \cdot z_1 + \underline{B}_1 \cdot \tilde{u}_1^{(k)}(h), \quad \dot{z}_2 = \underline{A}_2 \cdot z_2 + \underline{B}_2 \cdot \tilde{u}_2^{(j)}(h), \quad (4.11a)$$

$$y_1 = \underline{C}_1 \cdot z_1 + \underline{D}_1 \cdot \tilde{u}_1^{(k)}(h), \quad y_2 = \underline{C}_2 \cdot z_2 + \underline{D}_2 \cdot \tilde{u}_2^{(j)}(h). \quad (4.11b)$$

The subsystem equations are analytically solved for one macro step $T_N \rightarrow T_{N+1}$, or in step-size terms $h \in [0, H]$ where H is the constant macro-step size. The integrated states of the two subsystems are given by

$$\begin{aligned} z_1(h) &= \exp[\underline{A}_1 \cdot h] \cdot \left(\int_0^h \exp[-\underline{A}_1 \cdot \tau] \cdot \underline{B}_1 \cdot \tilde{u}_1^{(k)}(\tau) d\tau + z_{1,N} \right) \quad \text{with } z_{1,N} := z_1(h=0), \\ z_2(h) &= \exp[\underline{A}_2 \cdot h] \cdot \left(\int_0^h \exp[-\underline{A}_2 \cdot \tau] \cdot \underline{B}_2 \cdot \tilde{u}_2^{(j)}(\tau) d\tau + z_{2,N} \right) \quad \text{with } z_{2,N} := z_2(h=0). \end{aligned} \quad (4.12)$$

Evaluating the state solution (4.12) and the output Eqs. (4.11b) at $h = H$ and collecting the results in vectorized form, the variables $z_{N+1} := \begin{pmatrix} z_1(H) \\ z_2(H) \end{pmatrix}$ and

⁵The approximation error is only limited at 1e-12 due to round-off errors in the computer arithmetic.

$\underline{y}_{N+1} := \begin{pmatrix} y_1(H) \\ y_2(H) \end{pmatrix}$ at the end of the macro step are obtained. These variables are determined by a recurrence equation system of the form

$$\underbrace{\begin{pmatrix} \underline{z}_{N+1} \\ \underline{y}_{N+1} \\ \vdots \\ \underline{z}_{N+1-q} \\ \underline{y}_{N+1-q} \end{pmatrix}}_{\underline{Y}_{N+1}} = \underbrace{\begin{pmatrix} \underline{\tilde{\Omega}}_N & \underline{\tilde{\Omega}}_{N-1} & \cdots & \underline{\tilde{\Omega}}_{N-(q-1)} & \underline{\tilde{\Omega}}_{N-q} \\ \underline{I} & \underline{0} & \cdots & \underline{0} & \underline{0} \\ \underline{0} & \underline{I} & \ddots & \vdots & \vdots \\ \vdots & \ddots & \ddots & \underline{0} & \underline{0} \\ \underline{0} & \cdots & \underline{0} & \underline{I} & \underline{0} \end{pmatrix}}_{\underline{\Omega}} \cdot \underbrace{\begin{pmatrix} \underline{z}_N \\ \underline{y}_N \\ \vdots \\ \underline{z}_{N-q} \\ \underline{y}_{N-q} \end{pmatrix}}_{\underline{Y}_N} \quad (4.13)$$

where q is the order of the recurrence scheme which depends on the polynomial degrees k, j and the approximation method. The matrices $\underline{\tilde{\Omega}}_{N-i}$ as well as the companion matrix $\underline{\Omega}$ depend on the macro-step size and the model parameters.

The numerical stability of the applied co-simulation approach is computed by the spectral radius $\rho(\underline{\Omega})$ of the companion matrix. The method is called *unstable* if the magnitude of the largest eigenvalue satisfies the condition

$$\rho(\underline{\Omega}) := \|\lambda(\underline{\Omega})\|_\infty > 1 \quad , \quad (4.14)$$

which entails an exponentially increasing solution over time. Note that the time integration in the subsystems is carried out analytically in this analysis, so the indicator (4.14) only contains information about the instability of the numerical coupling and not about the instability of the subsystem solvers.

In Figs. 4.7, 4.8, 4.9, 4.10 and 4.11, the instability regions of the co-simulation approaches of Sect. 4.3 are shown, i.e. the configurations where $\rho(\underline{\Omega}) > 1$. The

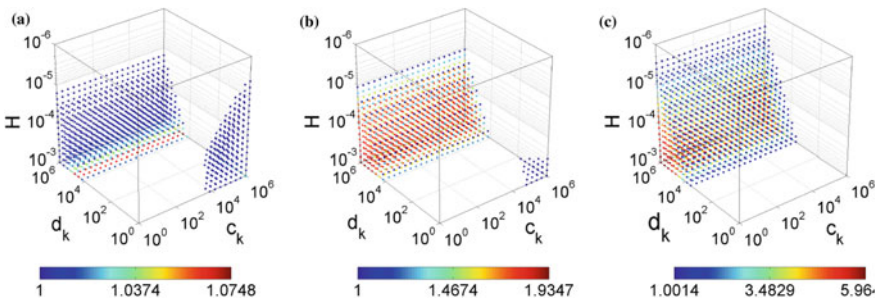


Fig. 4.7 Instability regions for a force/displacement coupling approach in combination with the sequential Gauss-Seidel scheme and a classical discontinuous extrapolation. Plot (a), (b) and (c) show the results for constant, linear and cubic polynomials

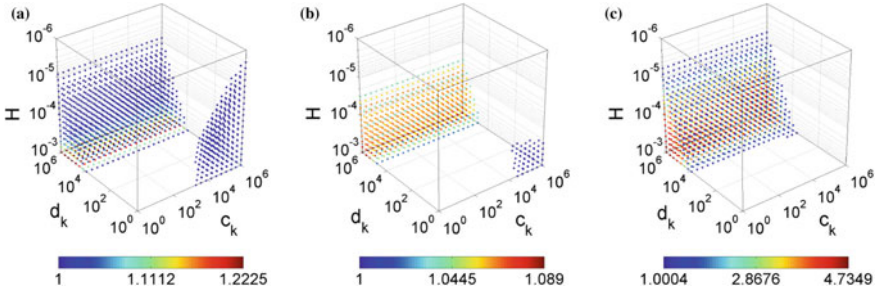


Fig. 4.8 Same plots as in Fig. 4.7, but the original C^0 -continuous EXTRIPOL method is applied

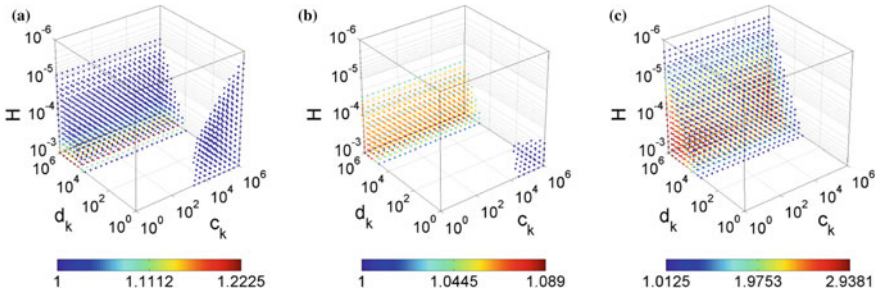


Fig. 4.9 Same plots as in Fig. 4.7, but the enhanced C^0 -continuous EXTRIPOL method is applied

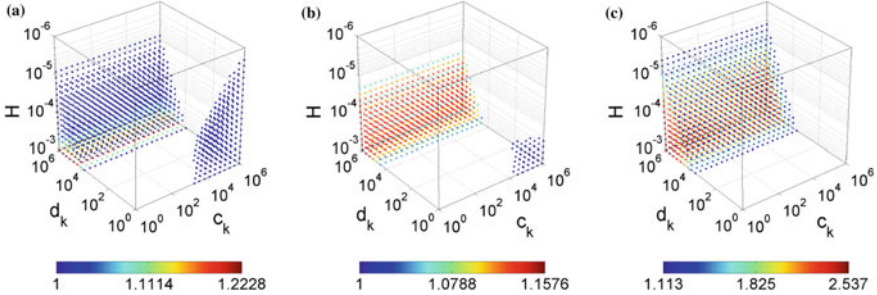


Fig. 4.10 Same plots as in Fig. 4.7, but the original C^1 -continuous EXTRIPOL method is applied

spectral radius ρ is plotted for a varying coupling stiffness c_k , a varying coupling-damping coefficient d_k and a varying macro-step size H in each plot. The remaining parameters of the test model are fixed according to $c_1 = 1E6$, $c_2 = 1E7$, $d_1 = 1$, $d_2 = 2$, $m_1 = 10$, $m_2 = 10$. The same polynomial degree p is used for the approximation in both subsystems, i.e. $k = j = p$, and the underlying extrapolation polynomials are chosen as constant ($p = 0$) in plot (a), linear ($p = 1$) in plot (b) and cubic ($p = 3$) in plot (c).

In each plots, unstable configurations can be found, e.g, for the use of high coupling-damping coefficients. However, all methods are zero-stable and the insta-

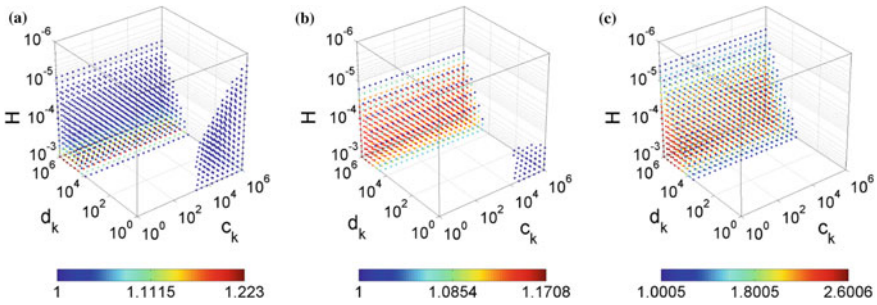


Fig. 4.11 Same plots as in Fig. 4.7, but the original C^2 -continuous EXTRIPOL method is applied

bility points vanish if the macro-step size H is reduced sufficiently. The instability regions for linear polynomials are smaller than the regions for constant polynomials which means that the co-simulation is stable at larger macro-step sizes. Applying cubic polynomials, the co-simulation gets more unstable again.

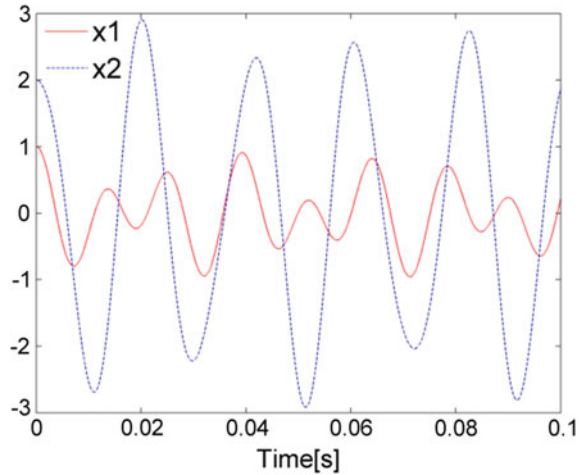
Comparing the original C^0 -continuous EXTRIPOL method in Fig. 4.8 with the classical approach in Fig. 4.7, the EXTRIPOL method is slightly more unstable for constant polynomials (a) and slightly more stable for linear and cubic polynomials (b and c). For instance, for a C^0 -continuous EXTRIPOL approach with a linear underlying polynomial, a 3-times larger macro-step size can be used as for the classical approach.

For the enhanced C^0 -continuous EXTRIPOL method (Fig. 4.9), the plots (a) and (b) are the same as for the original C^0 -continuous EXTRIPOL method (Fig. 4.8) since the approaches only differ for polynomial degrees $p > 1$. Using cubic polynomials, the enhanced method is more unstable than the original method and even slightly more unstable than the classical discontinuous method.

Also for the C^1 -continuous (Fig. 4.10) and the C^2 -continuous EXTRIPOL method (Fig. 4.11), only minor differences can be observed in the plots. For linear and cubic polynomials the methods are slightly more stable than the classical approach but slightly more unstable than the original C^0 -continuous EXTRIPOL method.

It can be concluded that all methods differ only in minor manner from scope of stability. This is surprising at the first sight since the EXTRIPOL methods and especially the enhanced C^0 -continuous method require more discrete coupling values for building the polynomials than the classical approach, see Sect. 4.3. However, in accordance with the results in Ref. [9], the stability behavior of numerical co-simulation methods is mainly determined by the degree of the polynomials and not by the amount of sampling points.

Fig. 4.12 Solution of test model for the error and performance investigation. x_1 and x_2 are the positions of the two masses



4.5 Global Error and Performance of the Co-simulation

In the following, it is analyzed if the subsystem solvers indeed benefit from a continuous co-simulation method. Thereto, the time integration in the subsystems is carried out numerically. A BDF method with variable step size and variable order is used in both subsystems. The step size of the solvers is limited by the macro-step size according to $h_{max} = H/2$ to prevent the solvers from stepping over several macro windows. The co-simulation is accomplished with a master/slave approach where the solver of the first subsystem (the *master*) calls the solver of the second subsystem (the *slave*) as a subroutine. When the master reaches the next macro-time point, it triggers the slave subroutine and waits while the slave solves its equations (*blocking call*). After the slave is finished with the macro step, it transmits the coupling data and the master can proceed with the next macro step. With this implementation, the co-simulation can advantageously be carried out on a single process instance.⁶

The example model from Sect. 4.2 is simulated 0.1 s. The model parameters are chosen as $c_1 = 1E5$, $c_2 = 1E3$, $c_k = 1E4$, $d_1 = 1$, $d_2 = 0$, $d_k = 0$, $m_1 = 0.5$, $m_2 = 0.1$ so that several oscillations arise in the investigated time interval, see Fig. 4.12. In Table 4.1, some results of the classical discontinuous co-simulation method are shown. The number of right-hand side calls of the master solver (*#RHS*) depends on the chosen macro-step size and on the tolerance. The number of communications (*#COM*) is mainly determined by the macro-step size. Please note that in the current implementation, the master solver is not forced to meet the macro-time points exactly. The slave subroutine is triggered as soon as the master solver steps over the macro-

⁶For a classical co-simulation, where both subsystems are solved on different process instances, the coupling has to be accomplished with inter-process communication which is more difficult to implement. The numerical approximation methods can however be applied in the same way.

Table 4.1 Simulation results of the classical discontinuous co-simulation: The number of right-hand side calls of the master solver ($\#RHS$), the number of macro steps ($\#COM$) and the global error (GE). The results are shown for several configurations of macro-step sizes H and solver tolerances Tol . Constant polynomials are used for the approximation ($p = 0$)

H	Tol	p	$\#RHS$	$\#COM$	GE
$1.0E - 03$	$1.0E - 04$	0	2703	88	$2.35E - 01$
$1.0E - 03$	$1.0E - 08$	0	12,312	95	$2.10E - 01$
$5.0E - 05$	$1.0E - 04$	0	13,288	1768	$6.97E - 03$
$5.0E - 05$	$1.0E - 08$	0	101,723	1685	$5.61E - 03$

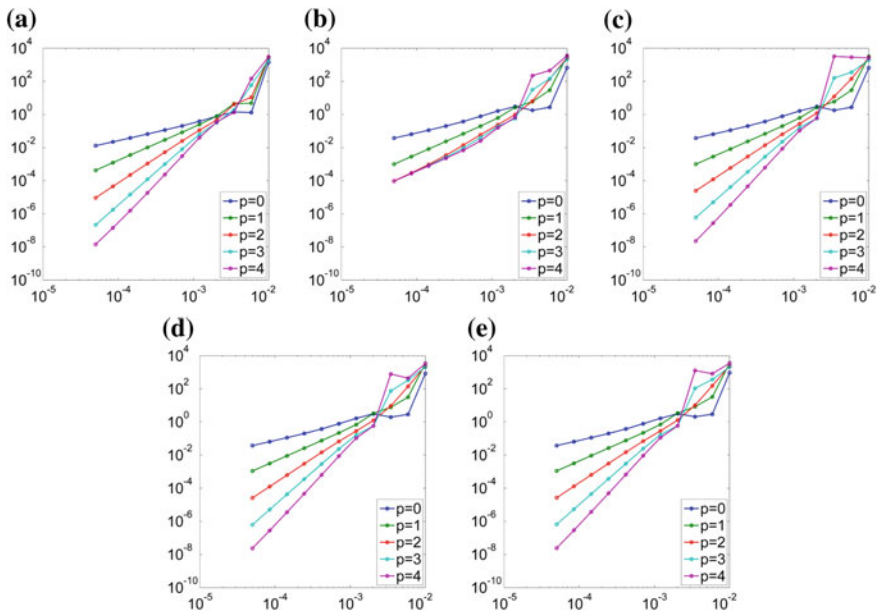


Fig. 4.13 Global error at the end of the simulation, shown logarithmically over the macro-step size H . The polynomial degrees are varied according to $p = 0, 1, 2, 3, 4$. The error is plotted for the classical discontinuous approach in (a), the original C^0 -continuous EXTRIPOL method in (b), the enhanced C^0 -continuous EXTRIPOL method in (c), the C^1 -continuous EXTRIPOL method in (d) and the C^2 -continuous EXTRIPOL method in (e)

time points at first. Consequently, the amount of communications is also slightly influenced by the solver tolerance.

In Fig. 4.13, the global error of the different co-simulation approaches is shown over the macro-step size H . The global error is calculated as the difference between the co-simulation solution and the analytical solution of the test model, evaluated at the end of the simulation. To investigate the asymptotic behavior of the co-simulation error, the solver tolerances are chosen comparatively small ($Tol = 1e - 10$). In each plot the polynomial degrees are varied between constant and quartic type

($p = 0, 1, 2, 3, 4$). Plot (a) shows the results for the classical discontinuous method, plots (b) and (c) for the original and the enhanced C^0 -continuous EXTRIPOL method and plots (d) and (e) for the C^1 and the C^2 -continuous EXTRIPOL method.

It can be observed that all methods converge for decreasing H which results from the fact that the methods are zero-stable and the approximation error decreases with H , see Sects. 4.4 and 4.3. The higher the polynomial degree is chosen, the faster the error is reduced. However, the error of the original C^0 -continuous EXTRIPOL method shows a limitation of the order which correlates with the order drop in the approximation error from Fig. 4.6f. This order limitation arises for polynomial degrees $p > 1$. In contrast, the enhanced C^0 -continuous EXTRIPOL method as well as the C^1 and the C^2 -continuous EXTRIPOL method generate the same error order as the classical discontinuous approach.

In Fig. 4.14, the performance of the EXTRIPOL methods is compared to the classical discontinuous approach. The performance is calculated in percent as the ratio

$$Performance := \left(\frac{\#RHS(EXTRIPOL)}{\#RHS(Classical)} - 1 \right) * 100 \quad (4.15)$$

of the number of solver right-hand side calls of the EXTRIPOL method and the number of right-hand side calls of the classical method. The performance is visualized as the color in Fig. 4.14. Blue or red color mean that the EXTRIPOL method is performing better or worse than the classical method. The plots show the performance for a varying macro-step size H and for a varying tolerance Tol of the BDF solver in the subsystems. For the sake of completeness, the tolerance and the macro-step size are varied in quite large ranges. The black rectangles in the plots indicate configurations which are commonly used in co-simulation application cases. Thus, only the results from the inner side of this rectangle might be of practical relevance.

The first row of the plot matrix shows the results for the original C^0 -continuous EXTRIPOL method (plots (a)–(c)), the second row for the enhanced C^0 -continuous EXTRIPOL method ((d)–(f)), the third row for the C^1 -continuous EXTRIPOL method ((g)–(i)) and the fourth row for the C^2 -continuous EXTRIPOL method ((j)–(l)). In the first, second and third column of the plot matrix, the results are shown for constant ($p = 0$), linear ($p = 1$) and cubic underlying polynomials ($p = 3$). The classical discontinuous approach is not listed in the figure since it is used as reference in Eq. (4.15), but some selected results of the classical approach can be found in Table 4.1 where the configurations of the corner points of the black rectangles were used in combination with a constant polynomial.

In general, it can be observed in the plots of Fig. 4.14 that the use of continuous approximation methods can be very advantageous for the performance of the co-simulation. In the dark-blue areas, the performance of the EXTRIPOL methods is larger than 50% which means an improvement of calculation effort by factor 2 or more. Further, the results confirm that the methods of higher continuity are mostly more advantageous for the performance than the C^0 -continuous approaches.

The performance benefit of the EXTRIPOL methods is especially large at constant and linear polynomials. At cubic polynomials, the blue regions are generally smaller.

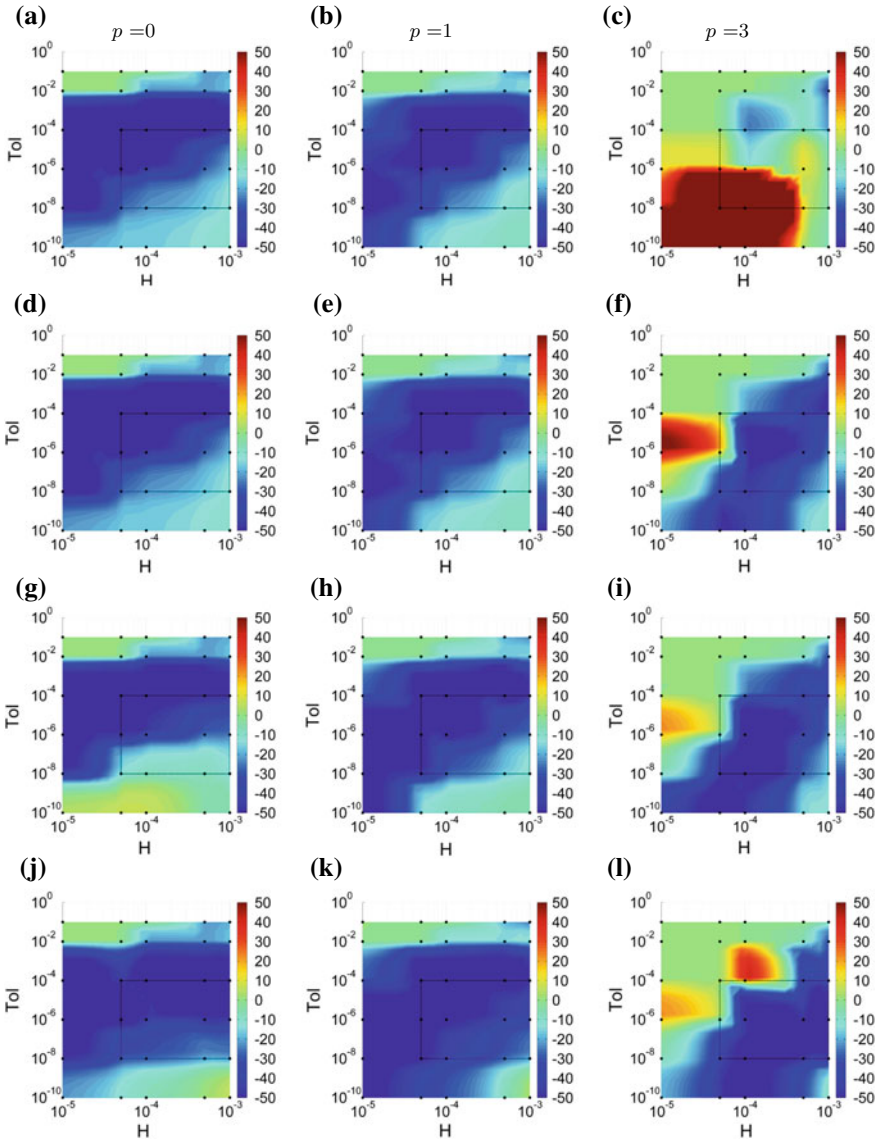


Fig. 4.14 Performance of the co-simulation if EXTRIPOL methods are used instead of the classical approach. The performance is shown in [%] for varying macro-step size H and solver tolerance Tol in each plot. The four rows of the plot matrix show the results for the original C^0 -continuous approach ((a)–(c)), for the enhanced C^0 -continuous approach ((d)–(f)), for the C^1 -continuous approach ((g)–(i)) and for the C^2 -continuous approach ((j)–(l)). The three columns of the plot matrix show the results for constant, linear and cubic polynomials. An implicit BDF solver is used in the subsystems. The black rectangles indicate common configurations of macro-step sizes and solver tolerances

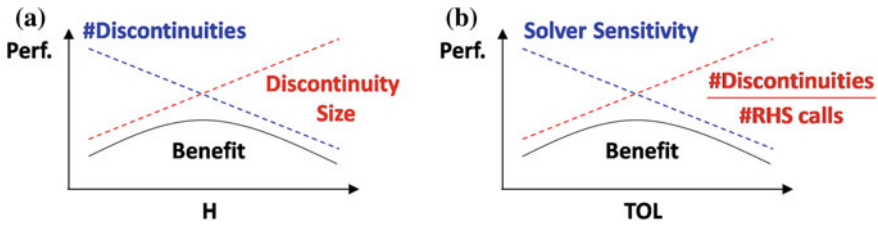


Fig. 4.15 Visualization of correlation between performance and macro-step size H (a) and between performance and solver tolerance TOL (b)

The original C^0 -continuous method even completely fails due to the limitation of the error order for $p > 1$. The order-preserving C^0 -continuous method and the methods with higher continuity perform much better. There are occasional configurations where also the C^1 and the C^2 -continuous method perform worse than the classical approach, but these configurations are mainly located outside of the black rectangle and are hence rarely applied in common co-simulation applications.

The plots in Fig. 4.14 share that the highest performance is mostly located in the middle of the plots. At the margins, the benefit is smaller. The reason for this behavior is visualized in Fig. 4.15. As the macro-step size H increases, see Fig. 4.15a, large-sized discontinuities are generated by the classical method (red line) and one would expect a high benefit with the continuous methods. However, a large macro-step size leads to a small amount of discontinuities (blue line) since the solvers interchange their data less often. Thus, the benefit (black curve) is limited. On the other hand, if H is small, the classical method generates a large amount of discontinuities, but the size of the discontinuities is small so that the solvers are less disturbed by the classical approach and the benefit of the continuous methods reduces. Hence, the benefit is maximized for medium macro-step sizes.

Similarly, the performance is correlated with the solver tolerance, see Fig. 4.15b. If the tolerance is small, the solvers are very sensitive to discontinuities (blue line), thus, one would expect a high benefit with the continuous methods. However, small tolerances force the solvers to make many right-hand side calls in general and the additional solver steps due to the discontinuities are of little significance for the performance, i.e., the ratio of the amount of discontinuities and RHS calls is small (red line). On the other hand, if the solver tolerance is large, the number of RHS calls is small and the additional solver steps due to the discontinuities would strongly determine the performance. However, the solvers are less sensitive at large tolerances and tolerate the discontinuities. Again, the performance is limited and the maximum benefit is obtained for medium solver tolerances.

Both correlations also depend on each other. To improve the benefit of the continuous methods, a simple reduction of the macro-step size is not sufficient. Additionally, the solver tolerance has to be reduced. Further, the location of the maximum benefit is influenced by the polynomial degree, as can be seen in the plots of Fig. 4.14.

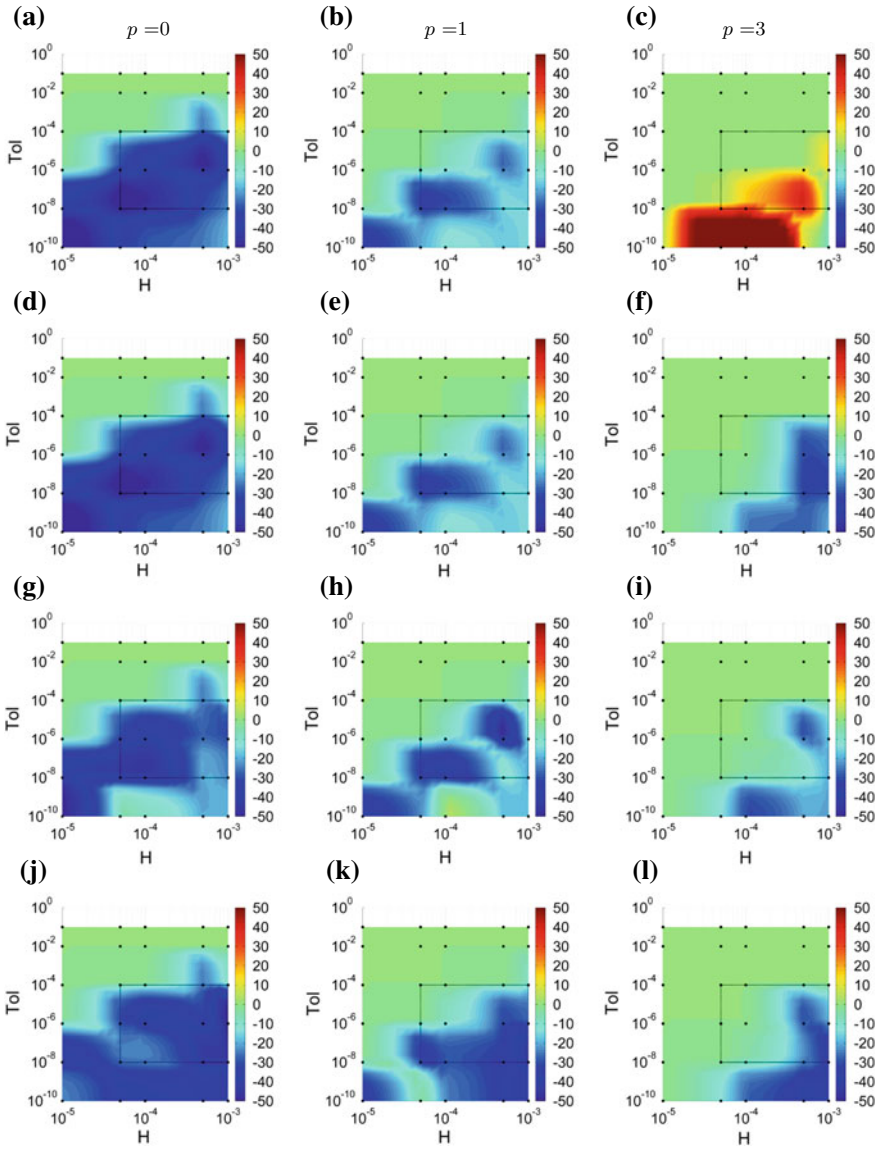


Fig. 4.16 Same plots as in Fig. 4.14, but a 4th-order Runge–Kutta method is used as solver

In Fig. 4.16, the subsystem solvers are replaced by an explicit Runge–Kutta method of order 4. The aforementioned conclusions are still valid. However, comparing the results with Fig. 4.14, the benefit of the continuous approaches is much smaller in combination with the explicit solver. Similar results are obtained if an explicit Adams method is used for solving the subsystems (results are not shown here). Explicit time integration methods do not require a computation of the Jacobian matrix in the solver steps and may be generally less sensitive to discontinuities in the right-hand side than implicit solvers.

4.6 Conclusion

Applying a continuous approximation method in co-simulation applications, the simulation performance can strongly be improved since the subsystem solvers are less disturbed by discontinuities at the macro-time points. A reduction of calculation time by factor 2 or more can be achieved for a wide range of co-simulation configurations. The performance benefit of the continuous methods mainly arises in combination with implicit subsystem solvers. With explicit solvers, the benefit is much smaller. Further, the benefit depends on the macro-step size H , the solver tolerance Tol and the degree p of the underlying extrapolation polynomials. Since these parameters have to be chosen problem-dependent, it should be checked for each simulation problem separately whether an EXTRIPOL method is of special benefit for the performance or whether the classical discontinuous method is sufficient.

In the current test, the C^2 -continuous EXTRIPOL method mostly outperformed the classical discontinuous approach and the EXTRIPOL methods of lower continuity. The original C^0 -continuous EXTRIPOL method fails in combination with super-linear extrapolation polynomials. The reason for this is a limitation of the error order which entails bad accuracy and bad performance. Using the enhanced C^0 -continuous approach, introduced in the present paper, or the C^1 and C^2 -continuous approach, the full convergence order of the classical discontinuous approach is preserved.

All considered EXTRIPOL methods are similar stable as the classical discontinuous approach and are especially zero-stable, as long as the subsystem solvers are zero-stable.

The C^1 and the C^2 -continuous method are based on Hermite polynomials. It is strongly recommended to use a Newton basis and divided differences for the implementation of these polynomials. Hermite polynomials in a Lagrange or in a monomial basis lead to an ill-conditioned interpolation problem and the numerical calculation results in singular Vandermonde matrices and large errors at small step sizes.

To achieve the full convergence order of the approximation methods, it is further necessary to fill up the polynomial sampling arrays at the beginning of the co-simulation (*initial calculation*). It is recommended to start the simulation with a sufficient amount of very small macro steps to increase the order of the polynomials. Then, the first normal-sized macro step can be completed with the high polynomial

order. Even though the polynomials are based on non-equidistant sampling points in the first macro steps, the stability of the co-simulation was not affected in the current test. As an advantage of the approach, no macro-step repetitions are required for the initial calculation and the explicit character of the co-simulation is maintained.

References

1. Ambrosio, J., Pombo, J., Rauter, F., Pereira, M.: A memory based communication in the co-simulation of multibody and finite element codes for pantograph-catenary interaction simulation. *Comput. Methods Appl. Sci.* **12**, 231–252 (2008)
2. Arnold, J., Einarsson, G., Krüger, W.: Multibody simulation of an oscillating aeroelastic wing model. *NAFEMS Int. J. CFD Case Stud.* **8**, 5–18 (2009)
3. Arnold, M.: Multi-rate time integration for large scale multibody system models. In: Eberhard, P. (ed.) *IUTAM Symposium on Multiscale Problems in Multibody System Contacts*, pp. 1–10. Springer, Dordrecht (2007)
4. Arnold, M.: Stability of sequential modular time integration methods for coupled multibody system models. *J. Comput. Nonlinear Dyn.* **5**, 031003 (2010). <https://doi.org/10.1115/1.4001389>
5. Arnold, M., Günther, M.: Preconditioned dynamic iteration for coupled differential-algebraic systems. *BIT Numer. Math.* **41**, 1–25 (2001)
6. Benedikt, M.: Eine Kopplungsmethode für die nicht-iterative Co-Simulation. Ph.D. thesis, TU Graz, Graz (2013)
7. Burger, M., Schneider, F., Steidel, S.: Coupled simulation in vehicle engineering. *PAMM* **16**, 493–494 (2016)
8. Busch, M.: Entwicklung einer SIMPACK-Modelica/Dymola Schnittstelle. Technical Report DLR-IB-515-07-02, DLR Deutsches Zentrum für Luft- und Raumfahrt e.V, Oberpfaffenhofen (2007)
9. Busch, M.: Zur effizienten Kopplung von Simulationsprogrammen (On the efficient coupling of simulation codes). Ph.D. thesis, University of Kassel, Kassel (2012). ISBN-13: 978-3862192960
10. Busch, M.: Continuous approximation techniques for co-simulation methods: Analysis of numerical stability and local error. *ZAMM - J. Appl. Math. Mech. - Zeitschrift für Angewandte Mathematik und Mechanik* **96**(9), 1061–1081 (2016)
11. Busch, M., Schweizer, B.: Co-simulation of multibody and finite-element systems: An efficient and robust semi-implicit coupling approach. *Arch. Appl. Mech.* **82**, 723–741 (2012)
12. Crow, M., Ilić, M.: The parallel implementation of the waveform relaxation method for transient stability simulations. *IEEE Trans. Power Syst.* **5**, 922–932 (1990)
13. Dietz, S., Hippmann, G., Schupp, G.: Interaction of vehicles and flexible tracks by co-simulation of multibody vehicle systems and finite element track models. In: True, H. (ed.) *The Dynamics of Vehicles on Roads and on Tracks*, Supplement to Vehicle System Dynamics, vol. 37, pp. 17–36. Swets & Zeitlinger, Lisse (2003)
14. Dronka, S., Rauh, J.: Co-simulation-interface for user-force-elements. In: *Proceedings of SIMPACK user meeting*. Baden-Baden (2006)
15. Feki, A.B.K.E., Duval, L., Faure, C., Simon, D., Gaid, M.B.: Choptrey: contextual online polynomial extrapolation for enhanced multi-core co-simulation of complex systems. *Simulation* **93**(3), 185–200 (2017)
16. Fleissner, F., Eberhard, P.: A Co-simulation approach for the 3D dynamic simulation of vehicles considering sloshing in cargo and fuel tanks. *PAMM* **9**, 133–134 (2009)
17. Friedrich, M., Ulbrich, H.: A parallel co-simulation for mechatronic systems. In: *Proceedings of the 1st Joint International Conference on Multibody System Dynamics*, pp. 1–10. Lappeenranta (2010)

18. Gail, T., Ober-Blöbaum, S., Leyendecker, S.: Variational multirate integration in discrete mechanics and optimal control. In: Proceedings of ECCOMAS 2017. Prague (2017)
19. Geusling, K., Bartel, A.: Coupling interfaces and their impact in field/circuit co-simulation. *IEEE Trans. Magn.* **53**(3), 1–4 (2016)
20. Gear, C.W., Wells, D.R.: Multirate linear multistep methods. *BIT Numer. Math.* **24**, 484–502 (1984)
21. Gomes, C., Karalis, P., Navarro-Lopez, E., Vangheluwe, H.: Approximated stability analysis of bi-modal hybrid co-simulation scenarios. In: Proceedings of 1st Workshop on Formal Co-Simulation of Cyber-Physical Systems. Trento, Italy (2017)
22. González, F., Naya, M., Luaces, A., González, M.: On the effect of multirate co-simulation techniques in the efficiency and accuracy of multibody system dynamics. *Multibody Syst. Dyn.* **25**(4), 461–483 (2011)
23. Gu, B., Asada, H.: Co-simulation of algebraically coupled dynamic subsystems without disclosure of proprietary subsystem models. *J. Dyn. Syst. Meas. Control* **126**, 1–13 (2004)
24. Knorr, S.: Multirateverfahren in der Co-Simulation. Master's thesis, Universität Ulm (2002)
25. Kübler, R., Schiehlen, W.: Two methods of simulator coupling. *Math. Comput. Model. Dyn. Syst.* **6**, 93–113 (2000)
26. Meyer, T., Schweizer, B.: Error estimation approach for controlling the communication step size for semi-implicit co-simulation methods. *PAMM* **15**(1), 63–64 (2015)
27. Park, K.C., Chiou, J.C., Downer, J.D.: Explicit-implicit staggered procedure for multibody dynamics analysis. *J. Guid. Control Dyn.* **13**, 562–570 (1990)
28. Savcenko, V.: Comparison of the asymptotic stability properties for two multirate strategies. *J. Comput. Appl. Math.* **220**, 508–524 (2008)
29. Schierz, T., Arnold, M., Clauß, C.: Co-simulation with communication step size control in an FMI compatible master algorithm. In: Proceedings of the 9th International Modelica Conference. Munich (2012)
30. Schweizer, B., Lu, D.: Semi-implicit co-simulation approach for solver coupling. *Arch. Appl. Mech.* **12**, 1739–1769 (2014)
31. Schweizer, B., Lu, D.: Stabilized index-2 co-simulation approach for solver coupling with algebraic constraints. *Multibody Syst. Dyn.* **34**, 129–161 (2015)
32. Song, C., Du, C., Li, G.: A unified modeling and parallel co-simulation method of cyber physical system. *Comput. Model. New Technol.* **17**(4), 217–223 (2013)
33. Vaculin, O., Krüger, W.R., Spieck, M.: Coupling of multibody and control simulation tools for the design of mechatronic systems. In: Proceedings of ASME 2001 International Design Engineering Technical Conferences. Pittsburgh (2001)
34. Valasek, M., Mraz, L.: Massive parallelization of multibody system simulation. *Acta Polytech.* **52**(6), 94–98 (2012)
35. Veitl, A., Arnold, M.: Coupled simulation of multibody systems and elastic structures. In: Ambrósio, J.A.C., Schiehlen, W. (eds.) *Advances in Computational Multibody Dynamics*, pp. 635–644. IDMEC/IST, Lisbon (1999)
36. Verhoeven, A., Tasič, B., Beelen, T.G.J., ter Maten, E.J.W., Mattheij, R.M.M.: Bdf compound-fast multirate transient analysis with adaptive step size control. *J. Numer. Anal. Ind. Appl. Math.* **1**, 1–3 (2007)
37. Wünsche, S., Clauß, C., Schwarz, P., Winkler, F.: Electro-thermal circuit simulation using simulator coupling. *IEEE Trans. Very Large Scale Integr. Syst.* **5**, 277–282 (1997)

Chapter 5

Stable Adaptive Co-simulation: A Switched Systems Approach



Cláudio Gomes, Benoît Legat, Raphaël M. Jungers and Hans Vangheluwe

Abstract Co-simulation promotes the idea that domain specific simulation tools should cooperate in order to simulate the inter-domain interactions that are often observed in complex systems. To get trustworthy results, it is important that this technique preserves the stability properties of the original system. In this paper, we show how to preserve stability for adaptive co-simulation schemes, which offer fine grained control over the performance/accuracy of the co-simulation. To this end, we apply the joint spectral radius theory to certify that an adaptive co-simulation scheme is stable, and, if that is not possible, we use recent results in this field to create a trace of decisions that lead to instability. With this trace, it is possible to adjust the adaptive co-simulation in order to make it stable. Our approach is limited by the fact that computing the joint spectral radius is NP-Hard and undecidable in general. Nevertheless, we successfully applied our results to the co-simulation of a double mass-spring-damper system.

C. Gomes (✉) · H. Vangheluwe
Department of Mathematics and Computer Science, University of Antwerp,
Antwerp, Belgium
e-mail: claudio.gomes@uantwerp.be

H. Vangheluwe
e-mail: hans.vangheluwe@uantwerp.be

B. Legat · R. M. Jungers
Université catholique de Louvain, ICTEAM, Ottignies-Louvain-la-Neuve, Belgium
e-mail: benoit.legat@uclouvain.be

R. M. Jungers
e-mail: raphael.jungers@uclouvain.be

C. Gomes · H. Vangheluwe
Flanders Make, Lommel, Belgium

H. Vangheluwe
School of Computer Science, McGill University, Montreal, QC, Canada

5.1 Introduction

Co-simulation is the simulation of a complex system via cooperating simulators, mimicking the interactions between subsystems [20, 24, 32, 40]. It promotes an efficient integration of the development process by leveraging existing, often specialized, modeling and simulation tools [9, 50], and can be applied at any stage [51]. Moreover, the parallelization and decoupling of the computation allows for faster simulations [8, 16, 36]. Here, simulator means any process that exhibits behavior over time, so this definition encompasses physical prototypes, software components, and human operators [3, 15, 19].

Throughout this paper we assume that the simulators are independent of each other.¹ As a consequence, an orchestrator is required to ensure that the simulators exchange data during a co-simulation.

Co-simulation promotes the idea that each simulator decides how to best compute the behavior of the subsystem allocated to it, leaving to the orchestrator the decision of when (with respect to the simulated time) should the simulators exchange data, and in what order [20]. However, as prior work has shown (e.g., [5, 6, 12, 17, 18, 29, 32, 45, 47]), the decision on how to best compute the behavior of each subsystem depends on the specific arrangement of all subsystems—such arrangement being called the co-simulation scenario—and on the decisions of the orchestrator. In sum: no decision concerning how to compute the co-simulation should be taken independently of the co-simulation scenario, which means that simulators should avoid “hard-coded” decisions.

It is currently a matter of research to find out which decisions are scenario dependent, and in this work, we assume that each simulator provides a mechanism to control some of these. Two factors are known to affect these decisions: (1) the co-simulation scenario; and (2) the requirements for the co-simulation.

Regarding the exact moment when these decisions need to be made, in the general case of systems that undergo structural changes (and therefore change the co-simulation scenario), the only possible time to make such decisions is when these changes occur, as demonstrated in [39]. The requirements for the co-simulation can change during the computation itself as well. The purpose of this is to inspect certain transient behavior of interest (e.g., see [7, 25, 30, 43]). We will therefore focus on adaptive co-simulation, where the orchestrator and simulators change the way they compute the co-simulation during the co-simulation itself, as a factors (1) and (2) change.

In the scope of adaptive co-simulation, it is hard to predict which decisions are to be taken without actually computing the co-simulation. It is then natural to wonder whether it is possible to ensure trustworthy co-simulation results, in the face of such uncertainty.

In this paper, we show how to prove that a co-simulation of a stable system is numerically stable, provided that the set of all possible decisions (i.e., reactions to

¹Two well known standards for co-simulation—the Functional Mockup Interface Standard for co-simulation [10], and the High Level Architecture [1]—share this assumption.

changes in factors (1) and (2)) is known. In particular, we propose to use the joint spectral radius theory [26] to certify the numerical stability of the co-simulation. Furthermore, when a co-simulation cannot be certified as numerically stable, we apply the results in [34, 35] to provide a numerically stable co-simulation with a reduced set of possible decisions.

The challenges associated with our approach lie in scaling with respect to the size of the underlying system, number of simulators, and number of decisions; and how to protect the Intellectual Property in each simulator.

In the next section, we introduce an example that motivates our research problem, and will serve as a running example. Section 5.3 presents some preliminary concepts related to stability in co-simulation and the techniques that we based our contribution on. Then, Sect. 5.4 details our contribution, and Sect. 5.5 the related work. Finally, Sect. 5.6 concludes.

5.2 Motivational Example

We motivate our work using a simple and well known system, that has been used to study the numerical stability of multiple orchestration algorithms (see, e.g., [4, 12–14, 28, 31, 45]).

A coupled mass-spring-damper system is shown in Fig. 5.1. We consider two simulators— S_1 , S_2 —and the allocation depicted in the figure: simulator S_1 computes the behavior of the left-hand-side (LHS) mass, accepting the input coupling force F_c , and producing the position and velocity of the mass as outputs; and S_2 accepts the position and velocity computed by S_1 , and produces the coupling force F_c . They are coupled as shown in Fig. 5.2.

Fig. 5.1 Example double mass-spring-damper system

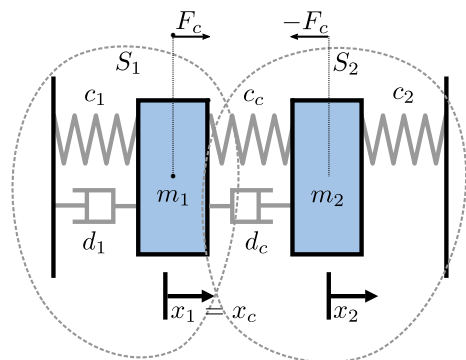
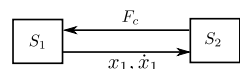


Fig. 5.2 Example arrangement of simulators



The dynamics of the LHS mass are given by:

$$\begin{aligned} \dot{x}_1(t) &= v_1(t); & m_1 \cdot \dot{v}_1(t) &= -c_1 \cdot x_1(t) - d_1 \cdot v_1(t) + F_c(t); \\ x_1(0) &= p_1; & v_1(0) &= s_1. \end{aligned} \quad (5.1)$$

where \dot{x} denotes the time derivative of x ; c_1 is the spring stiffness constant and d_1 the damping coefficient; m_1 is the mass; p_1 and s_1 the initial position and velocity, respectively; and $F_c(t)$ the input force acting on the mass over time.

The right-hand-side mass is governed by:

$$\begin{aligned} \dot{x}_2(t) &= v_2(t); & m_2 \cdot \dot{v}_2(t) &= -c_2 \cdot x_2(t) - F_c(t); \\ x_2(0) &= p_2; & v_2(0) &= s_2; \\ F_c(t) &= c_c \cdot (x_2(t) - x_1(t)) + d_c \cdot (v_2(t) - v_1(t)); \end{aligned} \quad (5.2)$$

where c_c and d_c denote the stiffness and damping coefficients of the central spring and damper, respectively; c_2 denotes the stiffness constant for the right spring; p_2 and s_2 the initial position and velocity.

We assume that the co-simulation of this example is computed as shown in Algorithm 1 (other orchestration algorithms exist—see [19, Sect. 4.2] for an overview). The function $\text{DOSTEP}(H, S)$ instructs simulator S to simulate the behavior of its allocated subsystem in the time interval $t \rightarrow t + H$. This computation is done using a numerical method and, since the input is not available in the open interval $(t, t + H)$, an extrapolation scheme is used.

Algorithm 1: Jacobi orchestration algorithm for the simulators shown in Fig. 5.2.

Data: The stop time t_f and a communication step size $H > 0$.

```

1  $t := 0$ ;
2 while  $t \leq t_f$  do
3    $[x_1 \ v_1]^T := \text{GETOUTPUT}(S_1)$ ;
4    $\text{SETINPUT}(S_2, [x_1 \ v_1]^T)$ ;
5    $F_c := \text{GETOUTPUT}(S_2)$ ;
6    $\text{SETINPUT}(S_1, F_c)$ ;
7    $\text{DOSTEP}(H, S_1)$ ;
8    $\text{DOSTEP}(H, S_2)$ ;
9    $t := t + H$ ;
10 end

```

Figure 5.3 shows multiple co-simulations of the system in Fig. 5.1, using different configurations for the simulators:

- x1 denotes the correct trajectory of $x_1(t)$, for reference, obtained by coupling Eqs. (5.1) and (5.2) and finding the analytical solution;
- x1_cs_1 denotes the trajectory obtained with a co-simulation where both simulators employ the forward Euler method, using a constant extrapolation of the inputs, and performing 10 internal integration steps per co-simulation step;

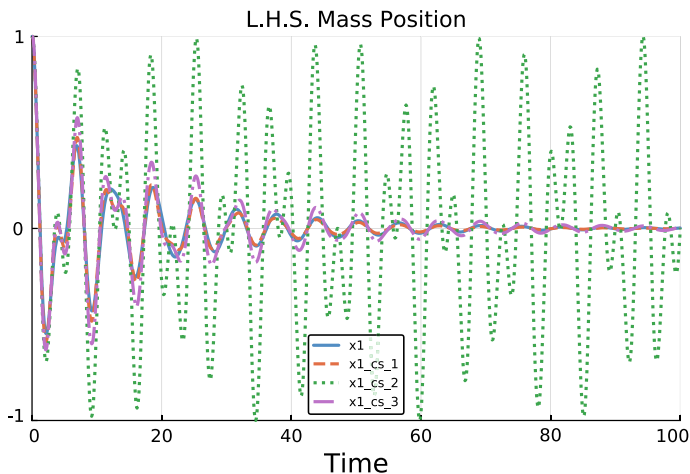


Fig. 5.3 LHS mass position co-simulations of the system in Fig. 5.1. Parameters: $m_1 = c_1 = m_2 = c_2 = c_c = 1.0$; and $d_1 = d_2 = d_c = 0.1$. The co-simulation step used is $H = 0.1$

$x1_cs_2$ is similar to $x1_cs_1$, except each simulator performs only one integration step per co-simulation step;

$x1_cs_3$ is obtained with a co-simulation that adaptively combines the configuration used in $x1_cs_1$ and $x1_cs_2$, i.e., it varies the number of internal integration steps per simulator.

Comparing the plotted trajectories, we see that there is something wrong with trajectory $x1_cs_2$. Due to the positive damping constants, the original system must always come to a rest, irrespective of the initial values. However, the co-simulation that produces $x1_cs_2$ does not seem to obey this law.

To compare the performance of each co-simulation, we compute the number of model evaluations. For the co-simulations producing the trajectories $x1_cs_1$ and $x1_cs_2$, this is given as:

$$\frac{t_f}{H} \times (\text{Steps}_{S_1} + \text{Steps}_{S_2}),$$

where t_f is the maximum simulation time, and Steps_S denotes the number of internal integration steps performed by simulator S , per invocation of $\text{DOSTEP}(H, S)$. The algorithm that computes trajectory $x1_cs_3$ is designed to spend 70% of the time using the parameters used to compute $x1_cs_1$ and the remaining time using the parameters used to compute $x1_cs_2$. It gives the following evaluations:

$$0.7 \times \text{Evals}_{CS_1} + 0.3 \times \text{Evals}_{CS_2}.$$

As can be seen in Table 5.1, the adaptive co-simulation mimics the qualitative behavior of the system (i.e., eventually coming to a rest), with fewer model evaluations than $x1_cs_1$.

Table 5.1 Total number of model evaluations per co-simulation in Fig. 5.3

Trajectory	Evaluations
x1_cs_1	20000
x1_cs_2	2000
x1_cs_3	14600

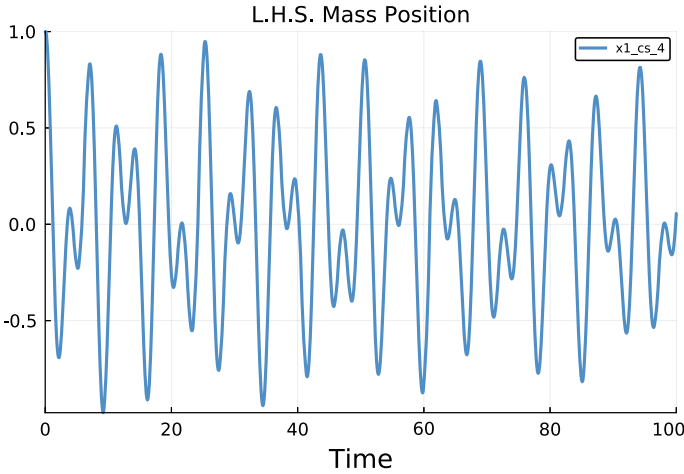


Fig. 5.4 Example wrong adaptive co-simulation

This minimal example highlights one of the advantages of adaptive co-simulations: the ability to obtain better tradeoffs between mimicking the qualitative behavior of the original system and performance.

Consider now the adaptive co-simulation $x1_cs_4$ shown in Fig. 5.4, which is similar to the policy used to compute $x1_cs_3$, except that more time is spent in the mode where the simulators only take one integration step. Despite being adaptive, it does not seem to come to a rest, which brings to our research problem: how can we tell a stable adaptive co-simulation, from an unstable one? And how can we ensure that the decisions taken during the co-simulation preserve the qualitative properties of the original system?

The next section provides the necessary background to explore this problem in depth.

5.3 Background

5.3.1 Co-simulation

In this paper, we consider a simulator to be an executable unit that expects input signals at agreed-upon points in simulated time, and produces output signals at these

times (see [19, Sect. 4] for a formal definition). The inputs (resp. outputs) correspond to the inputs (resp. outputs) of the subsystem that is allocated to that simulator.

A simulator often employs a numerical method to approximate the state of its subsystem over simulated time. Suppose that an input is provided at time t , and the simulator is instructed to compute until time $t + H$, with a given $H > 0$. Then, the simulator will perform a number of internal integration steps, while guessing what the input is throughout these steps. At time $t + H$, a new input point is provided, and the process is repeated.

A co-simulation scenario is a specific coupling of simulators, reflecting the coupling of the underlying subsystems. Here we assume that this coupling is a set of output-to-input assignments, giving rise to arrangements as exemplified in Fig. 5.2.

An orchestrator is an algorithm that uses the co-simulation scenario to compute a co-simulation. It is responsible for asking simulators to produce outputs, setting their inputs, and controlling their computation over the simulated time, by deciding the co-simulation policy. A simple orchestrator is shown in Algorithm 1.

A co-simulation policy is a sequence (over simulated time) of decisions that affect how the co-simulation is computed. In this paper, a policy encompasses:

Solver	the numerical solver used by each simulator;
Internal Step Size	the internal step size used by each simulator;
Input Approximation	the input extrapolation scheme used by each simulator; and
Orchestration	the order in which inputs are provided to each simulator (two well known examples are Jacobi and Gauss-Seidel orchestration [19]).

We consider these items because they are known to affect the stability of the co-simulation. For example, [12] studies the stability of the co-simulation using multiple input extrapolation schemes, and [45] studies the stability under different orchestration algorithms.

5.3.2 (Numerical) Stability

Consider the following initial value problem:

$$\dot{x} = Ax; \quad x(0) \text{ is given}; \quad (5.3)$$

where $x(t)$ is a real-valued vector, and A is a square real matrix.

The solution $x(t)$ to the system in Eq.(5.3) is stable if $x(t)$ tends to the origin, regardless of the initial value. In other words, $\lim_{t \rightarrow \infty} \|x(t)\| = 0$, for any given $x(0)$.

Suppose that the solution to Eq.(5.3) is approximated by the following discrete time system:

$$\tilde{x}_{i+1} = \tilde{A}\tilde{x}_i; \quad \tilde{x}_0 = x(0); \quad (5.4)$$

where \tilde{x}_i is a real-valued vector, and \tilde{A} is a square real matrix.

We say that the system in Eq. (5.4) is stable if, for all \tilde{x}_0 ,

$$\lim_{i \rightarrow \infty} \|\tilde{x}_i\| = 0 \Leftrightarrow \lim_{i \rightarrow \infty} \|\tilde{A}^i\| = 0, \quad (5.5)$$

for any vector norm $\|\tilde{x}\|$, and any matrix norm $\|\tilde{A}\|$ satisfying the *submultiplicativity* property.

Assuming that the system in Eq. (5.3) is stable, it is important that the approximating system in Eq. (5.4) preserves this property, in which case, we denote it as being numerically stable. The importance of preserving this property lies in the fact that the computation of the approximation naturally introduces errors. If the system in Eq. (5.4) is numerically stable, the errors introduced are not amplified.

The condition in Eq. (5.5) can be studied by means of the spectral radius $\rho(\tilde{A})$ [48, Theorem 1.3.2]:

$$\rho(\tilde{A}) < 1 \Leftrightarrow \lim_{i \rightarrow \infty} \|\tilde{A}^i\| = 0,$$

where $\rho(\tilde{A})$ is given by Gelfand's formula or the maximum absolute eigenvalue:

$$\rho(\tilde{A}) = \lim_{i \rightarrow \infty} \|\tilde{A}^i\|^{1/i} = \max_j |\lambda_j|, \quad (5.6)$$

and λ_j is the j th eigenvalue of \tilde{A} .

As detailed in [12, 19, 45] and references therein, the numerical stability of a co-simulation is analyzed by assuming that the underlying coupled system can be written as in Eq. (5.3) and is stable, and computing the discrete time system in the form of Eq. (5.4) that represents the co-simulation. Here, we illustrate how this is done for the example in Fig. 5.1 (a more general description is given in the above references). This procedure can be generalized to any number of simulators, as long as the underlying coupled system can be written as in Eq. (5.3) (for conditions that ensure this, see [5, Sect. 2]).

Consider now the example of Fig. 5.1, and suppose that the orchestrator (following Algorithm 1) and simulators are at time t_i . In the interval $t \in [t_i, t_{i+1}]$, each simulator S_j , with $j = 1, 2$, is trying to approximate the solution to a linear ODE,

$$\begin{aligned} \dot{x}_j &= A_j \cdot x_j + B_j \cdot u_j \\ y_j &= C_j \cdot x_j + D_j \cdot u_j \end{aligned} \quad (5.7)$$

where A_j, B_j, C_j, D_j are matrices with appropriate dimensions, and the initial state $x_j(t_i)$ is the state computed in the most recent co-simulation step. We assume that either D_1 or D_2 is the null matrix, so that the coupled system can be written as Eq. (5.3). In this example, $D_1 = \mathbf{0}$.

Without loss of generality (for more sophisticated input extrapolation techniques, see [12, Eq. (9)]), we assume that each simulator uses a constant extrapolation to approximate the input in the interval $[t_i, t_{i+1})$. That is, $\tilde{u}_j(t) = u_j(t_i)$, for $t \in [t_i, t_{i+1})$. Then, Eq. (5.7) can be re-written to represent the unforced system being integrated by each simulator:

$$\begin{bmatrix} \dot{x}_j \\ \dot{\tilde{u}}_j \end{bmatrix} = \begin{bmatrix} A_j & B_j \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \cdot \begin{bmatrix} x_j \\ \tilde{u}_j \end{bmatrix} \quad (5.8)$$

We can represent the multiple internal integration steps of Eq. (5.8), performed by the simulator S_j in the interval $t \in [t_i, t_{i+1}]$, as

$$\begin{bmatrix} x_j(t_{i+1}) \\ \tilde{u}_j(t_{i+1}) \end{bmatrix} = \tilde{A}_j^{k_j} \cdot \begin{bmatrix} x_j(t_i) \\ \tilde{u}_j \end{bmatrix} \quad (5.9)$$

where \tilde{A}_j represents a single integration step of the numerical method (e.g., $\tilde{A}_j = \mathbf{I} + h_j \begin{bmatrix} A_j & B_j \\ \mathbf{0} & \mathbf{0} \end{bmatrix}$ for the forward Euler method), $k_j = (t_{i+1} - t_i)/h_j$ is the number of internal steps, and $0 < h_j \leq H$ is the internal fixed step size that divides H . Note that Eq. (5.9) represents a discrete time system modeling the behavior of the simulator at a single co-simulation step, with no inputs. Now we just have to represent how the simulators exchange data at the end/beginning of a co-simulation step.

At the beginning of the co-simulation step i , we wish to enforce $u_1(t_i) = y_2(t_i)$ and $u_2(t_i) = y_1(t_i)$. This, together with Eq. (5.7), gives,

$$\begin{aligned} u_1(t_i) &= C_2 \cdot x_2(t_i) + D_2 C_1 \cdot x_1(t_i). \\ u_2(t_i) &= C_1 \cdot x_1(t_i) \end{aligned} \quad (5.10)$$

Finally, Eqs. (5.8)–(5.10) are combined to write the co-simulation step in the form of Eq. (5.4) as

$$\begin{bmatrix} x_1(t_{i+1}) \\ x_2(t_{i+1}) \end{bmatrix} = \underbrace{\begin{bmatrix} \mathbf{I} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \tilde{A}_1^{k_1} & \mathbf{0} \\ \mathbf{0} & \tilde{A}_2^{k_2} \end{bmatrix} \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & C_2 \\ \mathbf{0} & \mathbf{I} \\ C_1 & D_1 \cdot C_2 \end{bmatrix}}_{\tilde{A}} \begin{bmatrix} x_1(t_i) \\ x_2(t_i) \end{bmatrix} \quad (5.11)$$

whose stability is easily checked with Eq. (5.6).

We remark that Eq. (5.11) represents an abstraction of how the co-simulation is computed, for analysis purposes. In practice, the co-simulation itself may include optimizations, parallelism, etc ... which are neglected when building Eq. (5.11).

5.3.3 Joint Spectral Radius

The definitions we present here are adapted from [26].

Consider the following switched discrete time system:

$$x_{i+1} = A_{\sigma(i)} x_i : \sigma(i) \in \{0, \dots, m-1\}, A_{\sigma(i)} \in \Sigma \quad (5.12)$$

where x_0 is given, $\{0, \dots, m-1\}$ is the set of modes, $\sigma(i)$ is the mode active at step i , and $\Sigma = \{A_i\}_{i=0}^{m-1}$ is a sequence of m real square matrices.

We denote the sequence $\sigma(0), \sigma(1), \dots$ as the *switching signal*, where $A_{\sigma(i)} \in \Sigma$ represents the matrix used to compute x_{i+1} from x_i in Eq. (5.12). A switching signal $\sigma(0), \sigma(1), \dots, \sigma(i)$ induces the matrix product $A_{\sigma(i-1)} \dots A_{\sigma(1)} \cdot A_{\sigma(0)}$. Let

$$\Sigma^i = \{A_{p_{i-1}} A_{p_{i-2}} \dots A_{p_0} : A_{p_j} \in \Sigma, 0 \leq p_j < m, j = 0, \dots, i-1\}$$

be the set of all products induced by switching signals with length i . Note that, for any given switching signal $\sigma(0), \sigma(1), \dots, \sigma(i-1)$, $x_{i+1} = Ax_0$ for some $A \in \Sigma^i$.

The system in Eq. (5.12) is stable if, for any x_0 , and any switching signal, $\lim_{i \rightarrow \infty} \|x_i\| = 0$.

The Joint Spectral Radius $\hat{\rho}(\Sigma)$ (JSR) is essentially a generalization of Gelfand's formula, in Eq. (5.6), to arbitrary products of matrices in Σ [44]:

$$\begin{aligned} \hat{\rho}_i(\Sigma) &= \sup \{ \|A\|^{1/i} : A \in \Sigma^i \} \\ \hat{\rho}(\Sigma) &= \limsup_{i \rightarrow \infty} \hat{\rho}_i(\Sigma) \end{aligned} \tag{5.13}$$

Using the JSR, we can characterize the stability of the system in Eq. (5.12) by noting that [26, Theorem 1], for any bounded set Σ ,

$$\hat{\rho}(\Sigma) < 1 \Leftrightarrow \text{for all } \sigma, \lim_{i \rightarrow \infty} \|A_{\sigma(i)} A_{\sigma(i-1)} \dots A_{\sigma(0)}\| = 0. \tag{5.14}$$

To determine whether the system in Eq. (5.12) is stable, note that the limit in Eq. (5.13) exists, and any finite i satisfies:

$$\hat{\rho}(\Sigma) \leq \hat{\rho}_i(\Sigma) \text{ [26, Lemma 1.2].}$$

Therefore, if there exists i , such that $\hat{\rho}_i(\Sigma) < 1$, then the switched system is stable. Note however, that checking whether $\hat{\rho}(\Sigma) < 1$ is NP-Hard [11] and undecidable [26, Proposition 2.9] in general.

Other algorithms exist to estimate $\hat{\rho}(\Sigma)$, and we refer the reader to [21, 23, 27, 38, 42].

5.4 Stability Certification of Adaptive Co-simulations

In this section, we first describe how to use the concepts introduced in the previous section to determine the numerical stability of an adaptive co-simulation. Then, we propose a way to address the case when the adaptive co-simulation is not numerically stable.

5.4.1 Stability

Equation (5.11) represents a single co-simulation step, which, as can be seen from Eqs. (5.7) to (5.10), represents a specific: system arrangement; coupling approach; simulator input approximation; internal solver method; internal simulator step size h_j ; and communication step size H . If any of these items changes from one co-simulation step to the next, the co-simulation is adaptive, and is best described as a discrete time switched system, of the form of Eq. (5.12), where Σ includes every possible variation of the matrix A in Eq. (5.11), constructed as explained in Sect. 5.3.2.

To exemplify, in the co-simulation of the system in Fig. 5.1, suppose that the decision space is as follows:

Arrangement is the one in Fig. 5.2;

Coupling is the one in Algorithm 1 but a Gauss-Seidel, Strong coupling, or others, could have been used [22];

Input Approximation is the constant extrapolation but higher order input approximations can be applied [12];

Solver can be forward Euler, or the midpoint method [52, Sect. II.1];

Solver Step Size can be $H/10$ or H ;

Communication Step Size H is 0.1;

Then Σ contains 16 matrices, representing every possible combination of policies, per simulator, from one co-simulation step to the next.

Applying the result in Eq. (5.14) ensures that any possible decision sequence taken by the co-simulation always produces a numerically stable co-simulation. This is a strong result in the sense that we do not need to know anything about how the decisions are made.

In the example proposed, $\hat{\rho}(\Sigma) \geq 1$. To see why this is the case, let A_{cs_2} denote that co-simulation step matrix that uses $H = 0.1$ and solver step size equal to H . Then, computing the spectral radius $\rho(A)$, one observes that $\rho(A) > 1$. This means that there is a switching signal (always use A_{cs_2} to compute the next co-simulation step) that causes the co-simulation to not be stable. In fact, the result is the trajectory $x_{1_cs_2}$, plotted in Fig. 5.3.

5.4.2 Stabilization

As the paragraph above shows, if there is a matrix $A \in \Sigma$ such that $\rho(A) \geq 1$, then we have that $\hat{\rho}(\Sigma) \geq 1$. This immediately suggests an optimization to be done before computing the JSR: exclude all unstable matrices. That is, we set

$$\Sigma_0 = \Sigma \setminus \{A\}, \forall A \in \Sigma : \rho(A) \geq 1.$$

After computing Σ_0 , it can still be the case that $\hat{\rho}(\Sigma_0) \geq 1$, as the product of stable matrices is not necessarily stable (see, e.g., [26, Fig. 1.2]). Furthermore, $\hat{\rho}(\Sigma_0) \geq 1$

does not imply that there exists a finite i and a $A \in \Sigma_0^i$ such that $\rho(A) \geq 1$ (see, e.g., [26, Sect. 2.4], with the case that $\rho(A) = 1$). This means that no algorithm can always ensure that a stable co-simulation is attained. Fortunately, in practice, the algorithm proposed in [34] works well.

The work in [34] approximates $\hat{\rho}(\Sigma_0)$, allowing us to check whether $\hat{\rho}(\Sigma_0) < 1$, and, more importantly, returns a sequence p_0, \dots, p_{i-1} such that $\rho(A_{p_{i-1}} \dots A_{p_0}) \approx \hat{\rho}(M_0)$ to any desired level of accuracy. Computing $\Sigma_1 = \Sigma_0 \setminus A_{p_j}$ for one $j \in \{0, \dots, i-1\}$ and iterating allows one to obtain a Σ_* such that $\hat{\rho}(\Sigma_*) < 1$.

In the adaptive co-simulation of the system introduced in Fig. 5.3, we have that $\Sigma_* = \Sigma_0$ excludes the matrix A_{cs_2} , and $\hat{\rho}(\Sigma_0) \leq 0.992905$.

5.4.3 Conservativeness

As the previous result shows, applying this procedure to the adaptive co-simulation introduced in the previous sub-section results in a stable adaptive co-simulation that will never use the matrix A_{cs_2} . This is too restrictive. To see why, note that, as illustrated in plots of Fig. 5.3, a careful use of the decisions embedded in matrix A_{cs_2} actually yields a co-simulation that outperforms the other non-adaptive co-simulations (see the stable trajectory `x1_cs_3` in Table 5.1).

We propose a straightforward solution to this problem: apply the stabilization procedure to $Q = \Sigma^q$, which includes all products of length q of matrices in Σ for a given $q > 0$ (there is little use to including all products of length up to q because these are subsequences of the products of length m). The matrix products in the stabilized Q_* may include combinations of matrices that would otherwise have been removed.

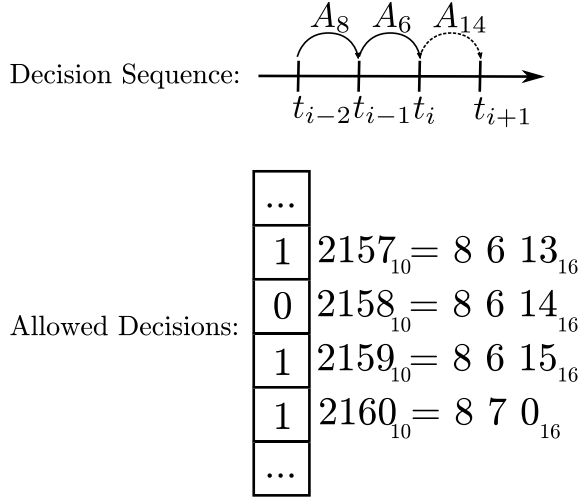
In the adaptive co-simulation example, we set $q = 2$ and we obtain Q_* that only excludes the matrix $A_{cs_2}A_{cs_2}$, which means that the policies embedded in A_{cs_2} can still be used, provided that they are alternated with any other policies. Applying the algorithm in [34] yields $\hat{\rho}(\Sigma_0^2) \leq 0.982986$.

5.4.4 Implementation

If the stabilization procedure terminates, we are left with Q_* : a set of sequences of matrix products of length q . Since we abstracted how each decision is taken at each co-simulation step, we still need to ensure that, at run-time, the decisions taken by the adaptive co-simulation remain within the allowed decisions (in the set Q_*).

To shed light on this problem, note that each sequence p_0, \dots, p_{q-1} that induces the matrix product $A_{p_{q-1}} \dots A_{p_0} \in \Sigma^q$, can be associated with one, and only one, natural number $d_{p_0 \dots p_{q-1}} \in \mathbb{N}_0$ computed as a conversion from base- m digit to a decimal number:

Fig. 5.5 Runtime structures of decision sequence monitor



$$d_{p_0 \dots p_{q-1}} = \sum_{j=0}^{q-1} p_j \cdot m^j, \tag{5.15}$$

where m is the number of matrices in Σ .

We therefore propose to allocate a m^q -bit array, where the position $d_{p_0 \dots p_{q-1}}$ of the array indicates whether the matrix product $A_{p_{q-1}} \dots A_{p_0} \in Q_*$. Then, at time t_i with $i \geq q - 1$, the previous q policies are used to reconstruct $d_{\sigma_{i-q-1} \dots \sigma_i}$ and check whether the corresponding subsequence is safe to take. If the policy $\sigma(i)$ is not safe to take, then the immediate neighbors of $d_{\sigma_{i-q-1} \dots \sigma_i}$ in the bit array can be inspected to find whether there are safe policies that can be selected. Figure 5.5 illustrates a scenario where $q = 3$, $m = 16$ and the orchestrator is about to decide to use the policies embedded in matrix A_{14} . A quick look-up to position 2158, obtained with Eq. (5.15), shows that this is not allowed. The neighbouring positions show alternative matrices that can be used.

5.5 Related Work

There is a huge body of work in techniques to determine the stability of a discrete time switched system. For introductions and overviews on the topic, please see [2, 26, 37, 41, 49].

We highlight the work in [33], where an algorithm is proposed to search for a stable periodic switching signal. This work differs from our because we are interested in removing all non-stable periodic switching signals, and retaining all the stable ones.

To the best of our knowledge, there is no work that applies the above results to the topic of adaptive co-simulation. For applications to the stability analysis of non-adaptive orchestration algorithms, we refer the reader to [12, 18, 45, 46].

5.6 Conclusion

In this paper, we introduce the problem of ensuring numerical stability in the context of adaptive co-simulation. To address this problem, we describe how to model the adaptive co-simulation as a discrete timed switched system, incorporating all possible policies. In particular, we describe a method to collect the information about each simulator and form a discrete time system (as in Eq. (5.11)), for each possible co-simulation configuration. Then we use recent results [34] to determine whether the adaptive co-simulation is numerically stable. If it is not, we propose an attempt to make it stable by reducing the set of allowed policies. Finally, we describe how to implement a simple monitor that ensures that the adaptive co-simulation only takes the accepted policies during execution.

The experiments made throughout the paper to exemplify our approach is available for download.²

There are three major limitations in this work: (i) the stability of the adaptive co-simulation is not decidable in general, so the algorithm we propose here may not terminate; (ii) for large numbers of policies, the computation of the joint spectral radius becomes impractical; (iii) in practice it may be hard to obtain the equations, solver, and input approximation technique, of each simulator, in order to form the system in Eq. (5.8). Another well known limitation, shared with traditional stability analysis techniques, is that our work is restricted to linear systems. However, for non-linear systems, linearization around an equilibrium point can be performed, in order to get some insights into the performance of the co-simulation at that equilibrium point.

To address limitation (ii), we will explore whether the construction of adaptive co-simulation yields any structure that can be leveraged (e.g., common reducibility) to accelerate the computation of the joint spectral radius. As for limitation (iii), we propose that each simulator discloses directly the matrix in Eq. (5.8), thereby avoiding the intellectual property issues.

References

1. IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA) - Federate Interface Specification. <https://standards.ieee.org/findstds/standard/1516-2010.html> (2010)
2. Ahmadi, A.A., Jungers, R., Parrilo, P.A., Roozbehani, M.: Analysis of the joint spectral radius via lyapunov functions on path-complete graphs. In: Proceedings of the 14th International Conference on Hybrid Systems: Computation and Control - HSCC '11, 13 pp. ACM Press, Chicago, IL, USA (2011). <https://doi.org/10.1145/1967701.1967706>

²http://msdl.cs.mcgill.ca/people/claudio/hybrid_cosim_analysis.

3. Alvarez Cabrera, A.A., Woestenenk, K., Tomiyama, T.: An architecture model to support cooperative design for mechatronic products: a control design case. *Mechatronics* **21**(3), 534–547 (2011). <https://doi.org/10.1016/j.mechatronics.2011.01.009>
4. Arnold, M.: Stability of sequential modular time integration methods for coupled multibody system models. *J. Comput. Nonlinear Dyn.* **5**(3), 9 (2010). <https://doi.org/10.1115/1.4001389>
5. Arnold, M., Clauß, C., Schierz, T.: Error analysis and error estimates for co-simulation in FMI for model exchange and co-simulation v2.0. In: Schöps, S., Bartel, A., Günther, M., ter Maten, W.E.J., Müller, C.P. (eds.) *Progress in Differential-Algebraic Equations*, pp. 107–125. Springer Berlin Heidelberg, Berlin, Heidelberg (2014). https://doi.org/10.1007/978-3-662-44926-4_6
6. Bastian, J., Clauß, C., Wolf, S., Schneider, P.: Master for co-simulation using FMI. In: 8th International Modelica Conference, pp. 115–120. Linköping University Electronic Press, Linköpings universitet, Dresden, Germany (2011). <https://doi.org/10.3384/ecp11063115>
7. Beltrame, G., Sciuto, D., Silvano, C.: Multi-accuracy power and performance transaction-level modeling. *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.* **26**(10), 1830–1842 (2007). <https://doi.org/10.1109/TCAD.2007.895790>
8. Ben Khaled, A., Ben Gaid, M., Pernet, N., Simon, D.: Fast multi-core co-simulation of cyber-physical systems: application to internal combustion engines. *Simul Model. Pract. Theory* **47**, 79–91 (2014). <https://doi.org/10.1016/j.simpat.2014.05.002>
9. Blochwitz, T., Otter, M., Arnold, M., Bausch, C., Clauss, C., Elmqvist, H., Junghanns, A., Mauss, J., Monteiro, M., Neidhold, T., Neumerkel, D., Olsson, H., Peetz, J.V., Wolf, S.: The Functional mockup interface for tool independent exchange of simulation models. In: 8th International Modelica Conference, pp. 105–114. Linköping University Electronic Press; Linköpings universitet, Dresden, Germany (2011). <https://doi.org/10.3384/ecp11063105>
10. Blockwitz, T., Otter, M., Akesson, J., Arnold, M., Clauss, C., Elmqvist, H., Friedrich, M., Junghanns, A., Mauss, J., Neumerkel, D., Olsson, H., Viel, A.: Functional mockup interface 2.0: the standard for tool independent exchange of simulation models. In: 9th International Modelica Conference, pp. 173–184. Linköping University Electronic Press, Munich, Germany (2012). <https://doi.org/10.3384/ecp12076173>
11. Blondel, V.D., Tsitsiklis, J.N.: The boundedness of all products of a pair of matrices is undecidable. *Syst. Control Lett.* **41**(2), 135–140 (2000). [https://doi.org/10.1016/S0167-6911\(00\)00049-9](https://doi.org/10.1016/S0167-6911(00)00049-9)
12. Busch, M.: Continuous approximation techniques for co-simulation methods: analysis of numerical stability and local error. *ZAMM J. Appl. Math. Mech.* **96**(9), 1061–1081 (2016). <https://doi.org/10.1002/zamm.201500196>
13. Busch, M., Schweizer, B.: Numerical stability and accuracy of different co-simulation techniques: analytical investigations based on a 2-DOF test model. In: 1st Joint International Conference on Multibody System Dynamics, pp. 25–27 (2010)
14. Busch, M., Schweizer, B.: Stability of co-simulation methods using Hermite and Lagrange approximation techniques. In: ECCOMAS Thematic Conference on Multibody Dynamics, Brussels, Belgium, pp. 1–10 (2011)
15. Faure, C., Ben Gaid, M., Pernet, N., Fremovici, M., Font, G., Corde, G.: Methods for real-time simulation of cyber-physical systems: application to automotive domain. In: 2011 7th International Wireless Communications and Mobile Computing Conference, pp. 1105–1110. IEEE (2011). <https://doi.org/10.1109/IWCMC.2011.5982695>
16. Friedrich, M.: Parallel co-simulation for mechatronic systems. Ph.D. thesis, Fakultät für Maschinenwesen (2011)
17. Gomes, C.: Foundations for continuous time hierarchical co-simulation. In: ACM Student Research Competition (ACM/IEEE 19th International Conference on Model Driven Engineering Languages and Systems), p. to appear. ACM, New York, USA, Saint Malo, France (2016)
18. Gomes, C., Karalis, P., Navarro-López, E.M., Vangheluwe, H.: Approximated stability analysis of bi-modal hybrid co-simulation scenarios. In: 1st Workshop on Formal Co-Simulation of Cyber-Physical Systems, pp. 345–360. Springer, Cham, Trento, Italy (2017). https://doi.org/10.1007/978-3-319-74781-1_24

19. Gomes, C., Thule, C., Broman, D., Larsen, P.G., Vangheluwe, H.: Co-simulation: state of the art. Technical report (2017). [arXiv:1702.00686](https://arxiv.org/abs/1702.00686)
20. Gomes, C., Thule, C., Broman, D., Larsen, P.G., Vangheluwe, H.: Co-simulation: a Survey. *ACM Comput. Surv.* **51**(3), Article 49 (2018). <https://doi.org/10.1145/3179993>
21. Gripenberg, G.: Computing the joint spectral radius. *Linear Algebra Appl.* **234**, 43–60 (1996). [https://doi.org/10.1016/0024-3795\(94\)00082-4](https://doi.org/10.1016/0024-3795(94)00082-4)
22. Gu, B., Asada, H.H.: Co-simulation of algebraically coupled dynamic subsystems. In: *American Control Conference*, vol. 3, pp. 2273–2278. IEEE, Arlington, VA, USA (2001). <https://doi.org/10.1109/ACC.2001.946089>
23. Guglielmi, N., Zennaro, M.: An algorithm for finding extremal polytope norms of matrix families. *Linear Algebra Appl.* **428**(10), 2265–2282 (2008). <https://doi.org/10.1016/j.laa.2007.07.009>, <http://linkinghub.elsevier.com/retrieve/pii/S0024379507003126>
24. Hafner, I., Popper, N.: On the terminology and structuring of co-simulation methods. In: *Proceedings of the 8th International Workshop on Equation-Based Object-Oriented Modeling Languages and Tools*, pp. 67–76. ACM Press, New York, USA (2017). <https://doi.org/10.1145/3158191.3158203>
25. Hines, K., Borriello, G.: Selective focus as a means of improving geographically distributed embedded system co-simulation. In: *8th IEEE International Workshop on Rapid System Prototyping*, pp. 58–62 (1997). <https://doi.org/10.1109/TWRSP.1997.618825>
26. Jungers, R.: *The Joint Spectral Radius: Theory and Applications*, vol. 385. Springer Science & Business Media, Berlin (2009)
27. Jungers, R.M., Cicone, A., Guglielmi, N.: Lifted polytope methods for computing the joint spectral radius. *SIAM J. Matrix Anal. Appl.* **35**(2), 391–410 (2014). <https://doi.org/10.1137/130907811>
28. Kalmar-Nagy, T., Stanculescu, I.: Can complex systems really be simulated? *Appl. Math. Comput.* **227**, 199–211 (2014). <https://doi.org/10.1016/j.amc.2013.11.037>
29. Karner, M., Steger, C., Weiss, R., Armengaud, E.: Optimizing HW/SW co-simulation based on run-time model switching. In: *2009 Forum on Specification & Design Languages, FDL*, pp. 1–6 (2009)
30. Karner, M., Armengaud, E., Steger, C., Weiss, R.: Holistic simulation of flexray networks by using run-time model switching. In: *Proceedings of the Conference on Design, Automation and Test in Europe, DATE '10*, pp. 544–549. European Design and Automation Association, 3001 Leuven, Belgium (2010)
31. Kübler, R., Schiehlen, W.: Modular simulation in multibody system dynamics. *Multibody Syst. Dyn.* **4**(2–3), 107–127 (2000). <https://doi.org/10.1023/A:1009810318420>
32. Kübler, R., Schiehlen, W.: Two methods of simulator coupling. *Math. Comput. Model. Dyn. Syst.* **6**(2), 93–113 (2000). [https://doi.org/10.1076/1387-3954\(200006\)6:2;1-M;FT093](https://doi.org/10.1076/1387-3954(200006)6:2;1-M;FT093)
33. Kundu, A., Chatterjee, D.: Stabilizing discrete-time switched linear systems. In: *Hybrid Systems: Computation and Control*, pp. 11–20. ACM Press, Berlin, Germany (2014). <https://doi.org/10.1145/2562059.2562114>, <http://dl.acm.org/citation.cfm?doi=2562059.2562114>
34. Legat, B., Jungers, R.M., Parrilo, P.A.: Generating unstable trajectories for switched systems via dual sum-of-squares techniques. In: *Proceedings of the 19th International Conference on Hybrid Systems: Computation and Control - HSCC '16*, pp. 51–60. ACM Press, New York, USA (2016). <https://doi.org/10.1145/2883817.2883821>
35. Legat, B., Parrilo, P.A., Jungers, R.M.: Certifying instability of switched systems using sum of squares programming. Technical report (2017). [arXiv:1710.01814](https://arxiv.org/abs/1710.01814)
36. Lelarsmee, E., Ruehli, A.E., Sangiovanni-Vincentelli, A.L.: The waveform relaxation method for time-domain analysis of large scale integrated circuits. *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.* **1**, 131–145 (1982). <https://doi.org/10.1109/TCAD.1982.1270004>
37. Lin, H., Antsaklis, P.J.: Stability and stabilizability of switched linear systems: a survey of recent results. *IEEE Trans. Autom. Control* **54**(2), 308–322 (2009). <https://doi.org/10.1109/TAC.2008.2012009>, <http://ieeexplore.ieee.org/document/4782010/>
38. Maesumi, M.: An efficient lower bound for the generalized spectral radius of a set of matrices. *Linear Algebra Appl.* **240**, 1–7 (1996). [https://doi.org/10.1016/0024-3795\(94\)00171-5](https://doi.org/10.1016/0024-3795(94)00171-5)

39. Mosterman, P.J.: An overview of hybrid simulation phenomena and their support by simulation packages. In: Vaandrager, F.W., van Schuppen J.H. (eds.) *Hybrid Systems: Computation and Control SE - 17*. Lecture Notes in Computer Science, vol. 1569, pp. 165–177. Springer Berlin Heidelberg, Berg en Dal, The Netherlands (1999). https://doi.org/10.1007/3-540-48983-5_17
40. Palensky, P., Van Der Meer, A.A., Lopez, C.D., Joseph, A., Pan, K.: Cosimulation of intelligent power systems: fundamentals, software architecture, numerics, and coupling. *IEEE Ind. Electron. Mag.* **11**(1), 34–50 (2017). <https://doi.org/10.1109/MIE.2016.2639825>, <http://ieeexplore.ieee.org/document/7883974/>
41. Parrilo, P.A., Jadbabaie, A.: Approximation of the joint spectral radius of a set of matrices using sum of squares. In: *Hybrid Systems: Computation and Control*, pp. 444–458. Springer, Berlin, Heidelberg, Pisa, Italy (2007). https://doi.org/10.1007/978-3-540-71493-4_35
42. Parrilo, P.A., Jadbabaie, A.: Approximation of the joint spectral radius using sum of squares. *Linear Algebra Appl.* **428**(10), 2385–2402 (2008). <https://doi.org/10.1016/j.laa.2007.12.027>, <http://www.sciencedirect.com/science/article/pii/S0024379508000281>, <http://linkinghub.elsevier.com/retrieve/pii/S0024379508000281>
43. Radetzki, M., Khaligh, R.S.: Accuracy-adaptive simulation of transaction level models. In: *Proceedings of the Conference on Design, Automation and Test in Europe, DATE '08*, pp. 788–791. ACM, New York, USA (2008). <https://doi.org/10.1145/1403375.1403566>
44. Rota, G.C., Strang, W.: A note on the joint spectral radius. In: *Proceedings of the Netherlands Academy*, vol. 22, pp. 379–381 (1960)
45. Schweizer, B., Li, P., Lu, D.: Explicit and implicit cosimulation methods: stability and convergence analysis for different solver coupling approaches. *J. Comput. Nonlinear Dyn.* **10**(5), 051,007 (2015). <https://doi.org/10.1115/1.4028503>
46. Schweizer, B., Li, P., Lu, D., Meyer, T.: Stabilized implicit co-simulation methods: solver coupling based on constitutive laws. *Arch. Appl. Mech.* **85**(11), 1559–1594 (2015). <https://doi.org/10.1007/s00419-015-0999-2>
47. Schweiger, G., Engel, G., Schoeggl, J., Hafner, I., Gomes, C., Nouidui, T.: Co-simulation – an empirical survey: applications, recent developments and future challenges. In: *MATHMOD 2018 Extended Abstract Volume*, pp. 125–126. ARGESIM Publisher Vienna, Vienna, Austria (2018). <https://doi.org/10.11128/arep.55.a55286>
48. Stuart, A., Humphries, A.R.: *Dynamical Systems and Numerical Analysis*, vol. 2. Cambridge University Press, Cambridge (1998)
49. Sun, Z., Ge, S.: Analysis and synthesis of switched linear control systems. *Automatica* **41**(2), 181–195 (2005). <https://doi.org/10.1016/j.automatica.2004.09.015>, <http://linkinghub.elsevier.com/retrieve/pii/S0005109804002778>
50. Tomiyama, T., D'Amelio, V., Urbanic, J., ElMaraghy, W.: Complexity of multi-disciplinary design. *CIRP Ann. Manuf. Technol.* **56**(1), 185–188 (2007). <https://doi.org/10.1016/j.cirp.2007.05.044>
51. Van der Auweraer, H., Anthonis, J., De Bruyne, S., Leuridan, J.: Virtual engineering at work: the challenges for designing mechatronic products. *Eng. Comput.* **29**(3), 389–408 (2013). <https://doi.org/10.1007/s00366-012-0286-6>
52. Wanner, G., Hairer, E.: *Solving Ordinary Differential Equations I: Nonstiff Problems*, vol. 1, Springer's edn. Springer, Berlin (1991)

Chapter 6

The SNiMoWrapper: An FMI-Compatible Testbed for Numerical Algorithms in Co-simulation



Stefan Hante, Martin Arnold and Markus Köbis

Abstract We introduce the SNiMoWrapper, an FMI-compatible software tool which enables the integration of models with an integrated, adapted solver in the form of a co-simulation FMU into simulation tools by conducting the co-simulation and hiding its details from the simulator. We describe the used algorithm in detail, give a short proof for the order of convergence of the SNiMoWrapper, show results for its application to an academic test example and describe an industrial proof-of-concept application.

6.1 Introduction

Co-simulation is a simulation technique for time-dependent coupled problems in engineering that restricts the data exchange between subsystems to discrete communication points in time. In the present paper we follow the block oriented framework in the industrial interface standard FMI for Model Exchange and Co-Simulation v2.0, see [6], and present the SNiMoWrapper, a testbed for numerical algorithms in co-simulation. It is designed to include FMI-compatible software components (Functional Mock-up Units or FMUs) in a simulation tool like MATLAB, Simulink or Simpack. We discuss a sophisticated implementation for hiding algorithmic details of co-simulation from the master simulation tool which does not even need to be aware that co-simulation is performed inside the SNiMoWrapper.

The paper is structured as follows: In Sect. 6.2, we will motivate the conception of the SNiMoWrapper and how it is supposed to work in a software environment. In Sect. 6.3 we will describe the co-simulation algorithm that is embedded into the SNiMoWrapper. We will discuss how the SNiMoWrapper was implemented in software in Sect. 6.4. In Sect. 6.5 we will analyze how the numerical errors behave, if the communication step size $H \rightarrow 0$ and present a proof of convergence of this algorithm, which is based on the convergence analysis in [1]. Section 6.6 describes some

S. Hante (✉) · M. Arnold · M. Köbis
Institute of Mathematics, Martin Luther University Halle-Wittenberg,
06099 Halle (Saale), Germany
e-mail: stefan.hante@mathematik.uni-halle.de

test-cases to which the SNIWrapper has been applied as well as some numerical tests that support the theoretical results from Sect. 6.5. Furthermore, we discuss an industrial proof-of-concept application of the SNIWrapper.

6.2 Approach

The Functional Mock-up Interface (FMI, [6]) is a powerful industrial standard that allows quick and easy import and export of software components in order to simulate complex physical structures that are composed of smaller sub-structures which themselves are driven by different physical concepts. A model of such a sub-structure is comprised inside a so-called Functional Mock-up Unit (FMU), often together with a time integration method specialized to the governing differential equations.

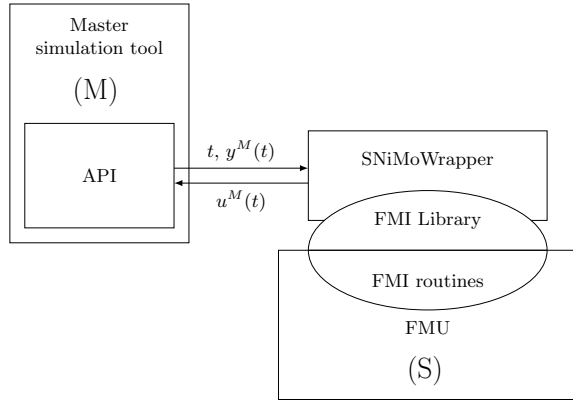
Model exchange FMUs give the importing simulation tool access to the internal states and right-hand side functions of the model, allowing the simulation tool to simply include the right-hand side calls for the model in its time integration routine and perform a monolithic time integration. This, however, is often unfavorable, because the dimension of the differential equation that has to be solved becomes large. Additionally, in this case the same time-integration method has to be applied to all the smaller submodels that may draw from different physical concepts. Often, for each submodel, there is a specialized numerical integration scheme that is known to work well with the type of model in question.

With co-simulation FMUs, this can be realized. The co-simulation FMU comes with an embedded numerical integration scheme and the importing simulation tool has no access to internal variables or right-hand side functions etc. This approach on the other hand needs an additional co-simulation algorithm that manages the data exchange between the different software modules. These topics are addressed by two standard approaches: The simulation backplane method, where there is a co-simulation code that manages all involved software modules and the master-driven method, where the co-simulation is handled by one specific simulation tool—the master simulation tool.

Our approach, however, is different: The whole co-simulation aspect is dealt with inside the SNIWrapper that is situated between the master simulation tool and the FMU that includes the so-called slave system. It effectively transforms the slave FMU into a function that can be called by the master simulation tool—with some restrictions—at any simulation time instance and therefore hides the details of the co-simulation from the master simulation tool.

The SNIWrapper is based on existing proprietary application programming interfaces (API) of industrial simulation software like m- or s-function in MATLAB/Simulink or user-defined force elements in industrial multi-body system simulation software. For a single master simulation tool an API front-end has to be developed whose only task is to communicate with the SNIWrapper. The SNIWrapper manages the co-simulation and communicates via FMI with the slave systems contained inside the FMU. The FMU comes with FMI v2.0 routines defined

Fig. 6.1 SNiMoWrapper’s workflow



in the standard [6], while the SNiMoWrapper uses a library to access these routines. Figure 6.1 shows workflow and basic software component of the SNiMoWrapper. The involved quantities $y^M(t)$ and $u^M(t)$ will be explained below.

In FMI for co-simulation, the data exchange between master and slave FMUs is restricted to discrete communication points T_i . During the communication step $(T_i, T_{i+1}]$ the slave systems are solved independently by their embedded solver [6]. The data exchange is handled by the SNiMoWrapper.

6.3 Co-simulation Algorithm

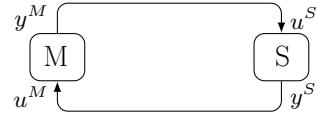
SNiMoWrapper’s co-simulation algorithm works with an equidistant communication point grid $T_i = T_0 + iH$ with constant communication step size $H > 0$.

Since, at this point, we want to couple a single co-simulation FMU with a master simulation tool, we have to consider two systems: The master system that represents the black-box system of the master simulation tool and the slave system that represents the co-simulation FMU. In the context of co-simulation, this block-oriented description was introduced in [9]. The coupled master-slave system is modeled mathematically by the following coupled set of differential equations

$$\left. \begin{aligned} \dot{x}^M(t) &= f^M(t, x^M(t), u^M(t)) \\ y^M(t) &= g^M(t, x^M(t), u^M(t)) \end{aligned} \right\} \quad (6.1)$$

$$\left. \begin{aligned} \dot{x}^S(t) &= f^S(t, x^S(t), u^S(t)) \\ y^S(t) &= g^S(t, x^S(t), u^S(t)) \end{aligned} \right\} \quad (6.2)$$

Fig. 6.2 Co-simulation setup of the SNIWoWrapper



where the coupling conditions are

$$\left. \begin{aligned} u^M(t) &= y^S(t) \\ u^S(t) &= y^M(t) \end{aligned} \right\}. \quad (6.3)$$

Here, x^M and x^S are internal states, u^M and u^S are inputs and y^M and y^S are outputs of the master and slave system, respectively. In Fig. 6.2, the black-boxes and their coupling is shown.

6.3.1 Prerequisites

In this paper, we are not dealing with coupled systems that have algebraic loops [1]. A system without direct feed-through is always free from algebraic loops, for instance.

The master tool may call the SNIWoWrapper through the API at any time instance t inside the current communication step $(T_i, T_{i+1}]$ in order to obtain $u^M(t)$. The master can call the SNIWoWrapper at any $T_i < t \leq T_{i+1}$ in any order but restricted to these bounds. This means, in particular, it may not go back to a previous communication step. The FMI standard does, in principle, allow for saving and reinitializing an FMU to a previous time instance. However, for many industrial models, it is a rather strong condition that the necessary routines `fmi2GetState`, `fmi2SetState` are implemented.

Before the master simulation tool can advance to the next communication step $(T_{i+1}, T_{i+2}]$, it has to call the SNIWoWrapper at T_{i+1} , so the SNIWoWrapper can obtain the master system's output $y^M(T_{i+1})$.

These prerequisites can typically easily be fulfilled by industrial simulation software.

6.3.2 Capabilities of the Slave System

The slave system is part of a co-simulation FMU, therefore there are only a few things that we can do with it. For co-simulation, FMI offers functions for setting slave inputs, performing time integration and retrieving slave outputs:

Setting inputs If the slave system is currently at time t^* , then we may set the inputs $u^S(t^*)$ at this time instance via the FMI function `fmi2SetReal`. If we were to perform a time integration of the slave system, then the inputs will be considered constant for the whole slave communication step. In order to provide inputs that vary in time, we will need to use the FMI function `fmi2SetRealInputDerivatives`. Using this, we can give a polynomial $Q(t)$ of maximum degree ρ to the slave system, that is represented by its derivatives at t^* (Nordsieck representation, see [6]) rather than by its coefficients:

$$Q(t) = \sum_{R=0}^{\rho} \frac{1}{k!} Q^{(R)}(t^*) (t - t^*)^R.$$

Via the FMI function `fmi2SetRealInputDerivatives`, we can give the polynomial's derivatives $Q^{(R)}(t^*)$ of order $R > 0$ to the slave system. Note that FMUs only support input derivatives up to a certain order `maxSlaveInputDerivatives`, which we denote by r . This effectively means that they can only handle inputs that are polynomials of degree up to $r = \text{maxSlaveInputDerivatives}$.

Performing time integration The time integration is triggered using the FMI function `fmi2DoStep`. It will run the time integration method of the FMU from $t^* \rightarrow t^* + H_S$ with a given slave communication step size $H_S > 0$, using the previously given inputs.

Getting outputs After the time integration was performed, we need to retrieve the outputs $y^S(t^* + H_S)$, where $t^* + H_S$ is the new current time of the slave system. This is done using the FMI function `fmi2GetReal`.

6.3.3 The Algorithm

The co-simulation algorithm of the SNiMoWrapper is a serial Gauss–Seidel co-simulation method involving higher order inter- and extrapolation of the inputs and outputs. We will present two variants of the algorithm: The basic algorithm and the extended algorithm. In the extended algorithm, we have put special attention on FMUs that do not support time-varying inputs, thus $r = \text{maxSlaveInputDerivatives} = 0$ as well as on FMUs that have highly-oscillating outputs. Pseudo code of the basic algorithm can be found in Algorithm 1 and of the extended algorithm in Algorithm 2. Furthermore, the extended algorithm is visualized in Fig. 6.3.

Note that due to the coupling conditions (6.3) we only use the terms y^S and y^M for the data that is stored inside the SNiMoWrapper. Throughout this paper we will use $p > 0$ as the order of interpolation of the master outputs y^M , $q > 0$ as the order of extrapolation of the slave outputs y^S and $r = \text{maxSlaveInputDerivatives}$ the maximal degree of polynomial that the slave can handle as input data. Also, we introduce the notation of the inter- and extrapolation polynomial π of maximum degree m in the following form

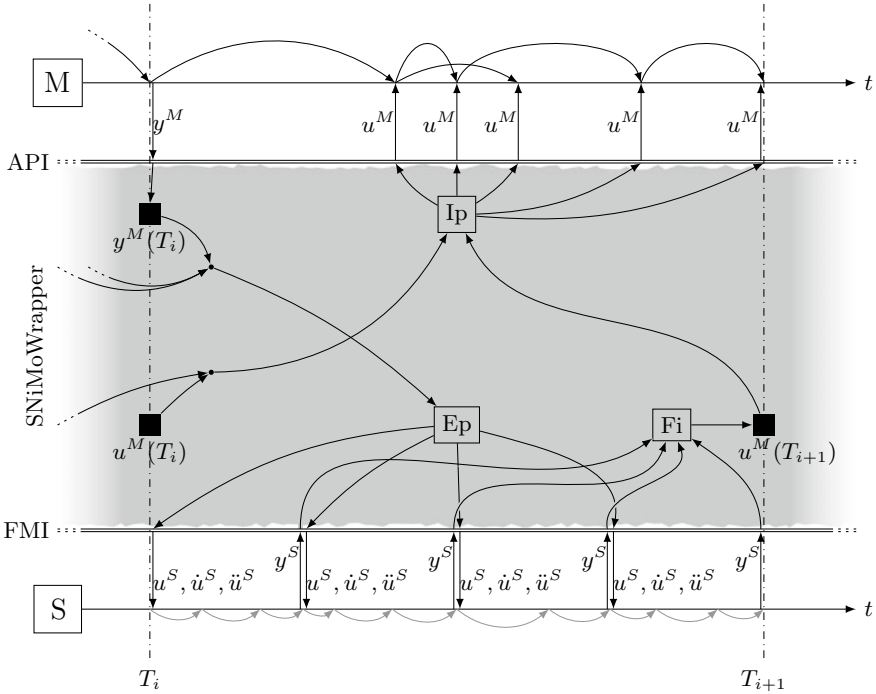


Fig. 6.3 Visualization of the SNiMoWrapper’s extended algorithm with $n = 4$ and $p = q = r = 2$. Ip stands for interpolation, Ep for extrapolation and Fi for filtering

$$\pi(\tau_j; \varphi; \tau_0, \dots, \tau_m) = \varphi(\tau_j), \quad (j = 0, \dots, m),$$

i.e., $\pi(\tau; \varphi; \tau_0, \dots, \tau_m)$ denotes the interpolation polynomial of a given function φ using the supporting points τ_0, \dots, τ_m . The maximum degree of the polynomial is determined by the number of arguments τ_0, \dots, τ_m .

The basic algorithm This variant of the algorithm is a well-known approach to co-simulation and uses higher order interpolation of the slave outputs and higher order extrapolation of the master outputs.

Calculating master inputs Assume that the master simulation tool requests input $u^M(t)$ with $t \in (T_i, T_{i+1}]$. The SNiMoWrapper will then interpolate the $p + 1$ master inputs $u^M(T_{i+1}), \dots, u^M(T_{i+1-p})$, that are already known to the SNiMoWrapper with an interpolation polynomial of maximum degree p :

$$u^M(t) = \pi(t; u^M; T_{i+1-p}, \dots, T_{i+1}).$$

The result of this calculation is then passed to the master simulation tool.

Calculating slave outputs When the master simulation tool requests input $u^M(t)$ with $t \in (T_i, T_{i+1}]$ for the first time, then $u^M(T_{i+1})$ has not been calculated yet. In order

to do so, we want to calculate the interpolation polynomial

$$u^S(t) = \pi(t; y^M; T_{i-q}, \dots, T_i)$$

of the $q + 1$ master outputs $y^M(T_i), \dots, y^M(T_{i-q})$, that have been passed to the SNiMoWrapper at the end of all preceding communication steps. This interpolation polynomial is then passed to the slave system in its Nordsieck form of degree r

$$\left((u^S)^{(R)}(T_i) \right)_{R=0}^r = \left(u^S(T_i), (u^S)'(T_i), \dots, (u^S)^{(r)}(T_i) \right),$$

because the slave only accepts polynomial of order up to r . Now the integration of the slave system from $T_i \rightarrow T_i + H = T_{i+1}$ is triggered. After the integration is finished, the master inputs at T_{i+1} are the slave outputs $u^M(T_{i+1}) := y^S(T_{i+1})$, which are retrieved and stored in the SNiMoWrapper.

As a variant, we can use linear interpolated extrapolation [5]

$$\begin{aligned} u^S(t) &= \pi_{ie}(t; y^M; T_i, T_{i-1}, T_{i-2}) \\ &= 2y^M(T_{i-1}) - y^M(T_{i-2}) + \frac{t - T_i}{H} (2y^M(T_i) - 3y^M(T_{i-1}) + y^M(T_{i-2})), \end{aligned}$$

that interpolates the results of two extrapolations at adjacent communication points. This approach will result in a continuous input signal [4, 5], but will restrict the order of the co-simulation algorithm to second order, since π_{ie} is a polynomial of maximum degree one, see Sect. 6.5 below.

In Algorithm 1 we have used the symbol $\tilde{\pi}$ to denote either the classical interpolation polynomial ($\tilde{\pi} = \pi$) or the interpolated extrapolation polynomial ($\tilde{\pi} = \pi_{ie}$).

Algorithm 1 Basic algorithm of the SNiMoWrapper

- 1: **if** $t = T_{i+1}$ **then** Save $y^M(T_{i+1})$
- 2: **if** $t > T_{i+1}$ **then**
- 3: $i \leftarrow i + 1$
- 4: Run slave from T_i to $T_i + H$ with input data $\left((u^S)^{(R)}(T_i) \right)_{R=0}^r$, where

$$u^S(\tau) = \tilde{\pi}(\tau; y^M; T_{i-q}, \dots, T_i)$$

- 5: Retrieve slave output $y^S(T_{i+1})$
 - 6: $u^M(T_{i+1}) := y^S(T_{i+1})$
 - 7: **return** $u^M(t) = \pi(t; u^M; T_{i-(p-1)}, \dots, T_i, T_{i+1})$
-

The extended algorithm Our extended algorithm is focused on FMUs that do not support higher order signal extrapolation, i.e. when $r = 0$ or very small, and on FMUs with highly oscillating behaviour, where even high order polynomial signal extrapolation can not be used to approximate the dynamics in a stable manner. The main idea is to introduce substeps in each master communication step $(T_i, T_{i+1}]$.

Throughout this paper let $n > 0$ be the number of substeps per master communication step and $H_S = H/n$ the slave communication step size.

The calculation of master inputs is done in exactly the same way, as in the basic algorithm. The main difference lies in the calculation of the slave outputs:

When the master simulation tool requests input $u^M(t)$ with $t \in (T_i, T_{i+1}]$ for the first time, as before, $u^M(T_{i+1})$ has not been calculated yet. Again, we consider the polynomial

$$u^S(t) = \tilde{\pi}(t; y^M; T_i, \dots, T_{i-q}),$$

which is either the classical interpolation polynomial ($\tilde{\pi} = \pi$) of maximum degree q or the linear interpolated extrapolation polynomial ($\tilde{\pi} = \pi_{ie}$). Now, for $k = 0, \dots, n-1$, we will recursively let the slave system integrate from $T_i + kH_S \rightarrow T_i + (k+1)H_S$ and afterwards retrieve the output $y^S(T_i + (k+1)H_S)$. The input data for each integration is the Nordsieck representation of order r of the polynomial $u^S(t)$ at the point $T_i + kH_S$:

$$\left((u^S)^{(R)}(T_i + kH_S) \right)_{R=0}^r.$$

After the slave system has reached the time instance $T_i + nH_S = T_i + H = T_{i+1}$, we can either use

$$u^M(T_{i+1}) := y^S(T_i + nH_S)$$

in order to continue the algorithm, or the retrieved slave outputs $y^S(T_i + kH_S)$ can be filtered. We have implemented the mean value filter

$$u^M(T_{i+1}) := \frac{1}{n} \sum_{k=1}^n y^S(T_i + kH_S)$$

in order to better support FMUs with highly oscillating outputs, where the high frequency oscillations are not relevant to the master system. In Algorithm 2 and Fig. 6.3 the filter function is denoted by the symbol Fi .

6.3.4 Initialization

The first call of the SNiMoWrapper should be at $t = T_0$ and the master tool is supposed to pass $y^M(T_0)$ to the SNiMoWrapper, in order to use it in its co-simulation algorithm. The SNiMoWrapper will then initialize the FMU and retrieve $y^S(T_0)$ from it. Since the algorithm needs y^S and y^M from previous communication points, we introduce the ghost communication points and ghost quantities

Algorithm 2 Extended algorithm of the SNIWoWrapper

-
- ```

1: if $t = T_{i+1}$ then Save $y^M(T_{i+1})$
2: if $t > T_{i+1}$ then
3: $i \leftarrow i + 1$
4: for $k = 0, \dots, n - 1$ do
5: Run slave from $T_i + kH_S$ to $T_i + (k + 1)H_S$ with input data

```

$$((u^S)^{(R)}(T_i + kH_S))_{R=0}^r,$$

where  $u^S(\tau) = \tilde{\pi}(\tau; y^M; T_{i-q}, \dots, T_i)$

- ```

6:   Retrieve slave output  $y^S(T_i + (k + 1)H_S)$ 
7:    $u^M(T_{i+1}) := \text{Fi}(y^S(T_i + H_S), \dots, y^S(T_i + nH_S))$ 
8: return  $u^M(t) = \pi(t; u^M; T_{i-(p-1)}, \dots, T_i, T_{i+1})$ 

```
-

$$\begin{aligned}
T_{-j} &:= T_0 - jH, & j &= 1, \dots, \max\{p - 1, q\} \\
y^S(T_{-j}) &:= y^S(T_0), & j &= 1, \dots, p - 1, \\
y^M(T_{-j}) &:= y^M(T_0), & j &= 1, \dots, q.
\end{aligned}$$

Although these starting values are only approximations of first order, for problems being dominated by time-dependent external excitations, the co-simulation algorithm still reaches its higher order. This can be seen in the numerical experiments in Sect. 6.6. For other problems the order of the algorithm will drop to first order if no higher-order initialization is used.

6.4 Implementation

The SNIWoWrapper was implemented in plain C due to the great portability of C code and the possibility to use the libraries listed below. The SNIWoWrapper library was compiled using the GNU C compiler from the GNU compiler collection (<https://gcc.gnu.org/>). We have tested the implementation on Windows as well as on Linux systems.

The SNIWoWrapper can be compiled into either a shared library, which then can be loaded by the master simulation tool, or can be compiled into a static library that can be linked into the simulation tool. Of course, the simulation tool has to be compatible with the compiler that was used to compile the SNIWoWrapper library.

For the incorporation of the FMI v2.0 routines, we used the open-source FMI Library FMIL from Modelon AB (<http://jmodelica.org/FMILibrary>). The setting of parameters for the SNIWoWrapper is handled by initialization files. In order to load and interpret the ini files, we have used the open-source library inih by Brush Technology (<http://code.google.com/p/inih/>).

The SNIWoWrapper is designed to support the handling of multiple FMUs, so the master simulation tool only needs to load the SNIWoWrapper once. This is done

using an ID that refers to a specific FMU. The usage of multiple instances of the same FMU is possible as well.

The APIs should be designed to give the SNIWoWrapper the time t at which the master system needs input data as well as the output data $y^M(t)$ at this time. If the master tool calls the SNIWoWrapper in a new communication step $(T_{i+1}, T_{i+2}]$, the SNIWoWrapper recognizes this and infers, that the last master output data must have been $y^M(T_{i+1})$.

We implemented a simple API for MATLAB, that uses MATLAB's capabilities to load external shared libraries. Based on this API, we implemented a custom Simulink block that enables the usage of the SNIWoWrapper in Simulink and acts as an API.

Furthermore, we implemented an API for the industrial multi-body system tool Simpact using the user-force-elements that are written in Fortran code. The SNIWoWrapper was compiled into a static library and linked to the compiled user-force-element code.

6.5 Accuracy

The convergence of co-simulation algorithms for systems without algebraic loops was studied in [1]. There it was assumed that the subsystems are solved with sufficiently small tolerances, so the analytic solution in each time-integration step is considered in order to concentrate on the co-simulation algorithm, following [2].

The result from [1] can be directly applied to the basic algorithm: The interpolation of the slave outputs is done with a polynomial of maximum degree q . The extrapolation of the master outputs is done with a polynomial of maximum degree p in the case of classical polynomial extrapolation and in the case of interpolated extrapolation the polynomial is at most linear. However, we can only give a polynomial of order r to the slave system. From this it follows that the overall interpolation error must be of order $\min\{q, p, r\} + 1$ in the polynomial extrapolation case and $\min\{q, 1, r\} + 1$ for interpolated extrapolation. From [1] it follows that the global error of the basic algorithm is of size $\mathcal{O}(H^{\min\{q, p, r\}+1})$ or $\mathcal{O}(H^{\min\{q, 1, r\}+1})$, respectively.

In the following, we will adapt the convergence analysis from [1] considering filtered slave output data and the substeps of the extended algorithm. Throughout the proof we will only use the inputs of the systems rather than the outputs, since they can easily be obtained by the coupling conditions (6.3). Let x^M, u^M, x^S and u^S be the analytic solutions of (6.1)–(6.3). The numerical solutions will be written with capital letters X^M, U^M, X^S, U^S and are for $t \in (T_i, T_{i+1}]$ the analytic solutions to

$$\dot{X}^M(t) = f^M(t, X^M(t), \Psi^M(t)), \quad (6.4a)$$

$$\Psi^M(t) = \pi(t; U^M; T_{i+1-p}, \dots, T_{i+1}), \quad (6.4b)$$

$$U^S(T_{i+1}) = g^M(X^M(T_{i+1}), U^M(T_{i+1})), \quad (6.4c)$$

$$\dot{X}^S(t) = f^S(t, X^S(t), \Psi^S(t)), \quad (6.4d)$$

$$\Phi(t) = \pi(t; U^S; T_{i-q}, \dots, T_i), \quad (6.4e)$$

$$\Psi^S(t) = \mathcal{T}_r(t; T_k + kH_S; \Phi), \quad \begin{cases} t \in (T_i + kH_S, T_i + (k+1)H_S], \\ k = 0, \dots, n-1, \end{cases} \quad (6.4f)$$

$$U^M(T_{i+1}) = \text{Fi} \left[(g^S(X^S(T_i + kH_S), \Psi^S(T_i + kH_S)))_{k=1}^n \right], \quad (6.4g)$$

where $\mathcal{T}_r(\tau; \tau^*; \varphi)$ is the Taylor polynomial of order r around τ^* of function φ evaluated at τ . This Taylor polynomial describes the truncation of the Nordsieck representation to order r . Because the functions Ψ^S and Ψ^M are fundamentally different, we will consider the components of the master and slave system separately.

First, we want to consider the error in the internal states for a step $T_i \rightarrow T_i + H$, so let $t \in (T_i, T_{i+1}]$. A perturbation analysis of the ordinary differential equations (6.4a) and (6.4d) together with their dependence on the initial value [13] gives for $B \in \{S, M\}$

$$\begin{aligned} \|X^B(T_{i+1}) - x^B(T_{i+1})\| &\leq e^{L_1 H} \|X^B(T_i) - x^B(T_i)\| \\ &\quad + L_2 \frac{e^{L_1 H} - 1}{L_1} \max_{t \in [T_i, T_{i+1}]} \|\Psi^B(t) - u^B(t)\| \end{aligned} \quad (6.5)$$

with some constants $L_1, L_2 > 0$. For the master system the last difference can be estimated by

$$\Psi^M(t) - u^M(t) = \mathcal{O}(1) \sum_{j=0}^p (U^M(T_{i+1-j}) - u^M(T_{i+1-j})) + \mathcal{O}(H^{p+1}), \quad (6.6)$$

because the interpolation polynomial is linear in its values of support. Furthermore, we need a standard result on the error of polynomial interpolation. For the slave system, we need to deal with the truncation to degree r :

$$\Psi^S(t) - u^S(t) = \mathcal{O}(1) \sum_{j=0}^q (U^M(T_{i-j}) - u^M(T_{i-j})) + \mathcal{O}(H^{q+1} + H_S^{r+1}). \quad (6.7)$$

Here, we have the term H_S^{r+1} that results from the fact that the truncation to degree r happens in an interval of length H_S . Of course, asymptotically it is $\mathcal{O}(H_S^{r+1}) = \mathcal{O}(H^{r+1})$, but the more careful way of writing H_S^{r+1} explains the results for coarse H and larger n and therefore smaller H_S in Test 2 in Sect. 6.6.

We introduce the following notation for the errors in the states and inputs for $B \in \{S, M\}$:

$$\begin{aligned} \epsilon_i^B &:= \|X^B(T_i) - x^B(T_i)\|, \\ \eta_i^B &:= \|U^B(T_i) - u^B(T_i)\|. \end{aligned}$$

From (6.4c) and (6.4g) it now follows

$$U^S(T_{i+1}) - u^S(T_{i+1}) = \mathcal{O}(1)\epsilon_{i+1}^M + J_{i+1}^M(U_{i+1}^M - u^M(T_{i+1})), \quad (6.8a)$$

$$U^M(T_{i+1}) - u^M(T_{i+1}) = \mathcal{O}(1)\epsilon_{i+1}^S + \mathcal{O}(1)J_{i+1}^S \sum_{j=0}^q (U_{i-j}^S - u^S(T_{i-j})) \\ + \mathcal{O}(H^{q+1} + H_S^{r+1} + F(H)), \quad (6.8b)$$

where J_{i+1}^M and J_{i+1}^S are Jacobians of $g^M = g^M(x^M, u^M)$ and $g^S = g^S(x^S, u^S)$ with respect to u^M and u^S , respectively. The term $F(H)$ represents the error of the filtering, that is applied in (6.4g). If there is no filtering, then we have $F(H) = 0$ and in the case of the mean value filter, we get $F(H) = H$, because the averaging is a first order approximation to each element.

Since the system is supposed to be free of algebraic loops, we can then plug the Eqs. (6.8a) and (6.8b) into each other repeatedly and end up with

$$\eta_{i+1}^M + \eta_{i+1}^S = \mathcal{O}(1) \sum_{j=0}^{m(q+1)-1} (\epsilon_{i-j}^S + \epsilon_{i-j}^M) + \mathcal{O}(H^{q+1} + H_S^{r+1} + F(H)) \quad (6.9)$$

for $B \in \{S, M\}$ and an $m > 0$ by further estimation [1]. Now it follows

$$\epsilon_{i+1}^M + \epsilon_{i+1}^S = (1 + \mathcal{O}(H))(\epsilon_i^M + \epsilon_i^S) + \mathcal{O}(H) \sum_{j=1}^{m \max\{p, q+1\}} (\eta_{k+1-j}^S + \eta_{k+1-j}^M) \\ + \mathcal{O}(H^{p+2} + H^{q+2} + H \cdot H_S^{r+1} + H \cdot F(H)). \quad (6.10)$$

The term $\mathcal{O}(H)(\epsilon_{k+1}^M + \epsilon_{k+1}^S)$ that would appear on the right-hand side can be brought to the left side. Dividing by $1 - \mathcal{O}(H) = \mathcal{O}(1)$ leads to (6.10). This analysis roughly followed the convergence analysis for linear multistep methods in the DAE case [8].

Now, just like in [1], it follows

$$\epsilon_{i+1}^M + \epsilon_{i+1}^S, \eta_{i+1}^M + \eta_{i+1}^S \in \mathcal{O}(H^{p+1} + H^{q+1} + H_S^{r+1} + F(H)).$$

This means that the extended algorithm is of order $\min\{p, q, r\} + 1$. In the case of interpolated extrapolation, the same argument holds and we get convergence of order $\min\{p, 1, r\} + 1$. If the filtering is applied, we get a convergence of first order.

The asymptotic analysis for $H \rightarrow 0$ is not insightful in the case of a highly oscillating slave system, where a filtering would be applied. Here, a convergence analysis in the frequency range would be more appropriate.

6.6 Applications

In the following, we will present applications of the SNiMoWrapper. Firstly, we have conducted numerical tests on an academic example in order to support the theoretical results of Sect. 6.5. Then, as a proof of concept we discuss two real-world problems in Sect. 6.6.2.

6.6.1 Academic Example

The test example is a quarter car model [12], see also [3]. The chassis and wheel are represented by two point masses m_M and m_S and can only move in the vertical direction. The chassis is connected to the wheel by a damper and a spring with coefficients d_M and k_M , while the wheel is connected to the ground by a very stiff spring with stiffness k_S . The master subsystem only consists of the chassis, while the slave subsystem consists of the wheel and the ground. We apply a force-displacement coupling. This means that the master subsystem outputs the height of the chassis and the slave subsystem outputs the force that is applied by the spring and damper that connect wheel and chassis. The situation is depicted in Fig. 6.4. The input of the master is, as above, the output of the slave and vice versa. In order to model a moving quarter car, the height of the ground $h(t)$ varies in time. For this, we have used two options:

$$\begin{aligned} \text{smooth profile: } h(t) &= \begin{cases} 0.1 \text{ m} \cdot \exp\left(\frac{1}{(t-2)^2-1}\right), & t \in (1 \text{ s}, 3 \text{ s}), \\ 0 \text{ m}, & \text{else,} \end{cases} \\ \text{step profile: } h(t) &= \begin{cases} 0.04 \text{ m}, & t < 4 \text{ s}, \\ 0 \text{ m}, & \text{else,} \end{cases} \end{aligned}$$

see Fig. 6.5. The model parameters are given by [10]:

$$\begin{aligned} m_M &= 256 \text{ kg}, & k_M &= 2020 \text{ N/m}, & d_M &= 1140 \text{ Ns/m}, \\ m_S &= 31 \text{ kg}, & k_S &= 128 \text{ kN/m}. \end{aligned}$$

The master subsystem was implemented in Fortran. For the time integration of the master subsystem we used DASSL [11], a popular Fortran implementation of the BDF multistep time integration method. The slave subsystem was implemented in an FMU using plain C. Here, the time integration method is DOPRI5 [7], an implementation of the Runge–Kutta time integration method of Dormand and Prince with fifth order.

In the following, we have conducted numerical experiments in order to verify the theoretical analysis and to illustrate the effects of some of the co-simulation options. All tests have been performed with tight absolute and relative tolerances of

Fig. 6.4 Quarter car model

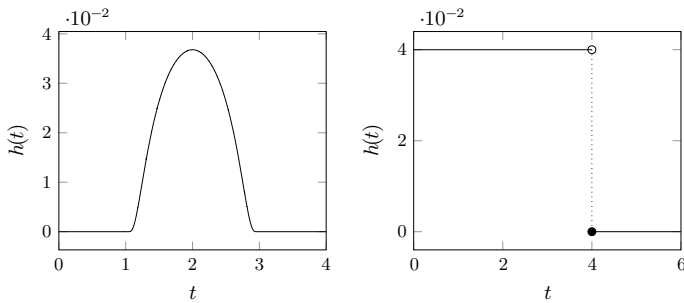
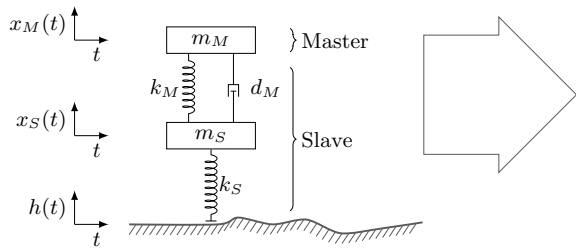


Fig. 6.5 Plot of the smooth road profile (left) and of the step road profile (right)

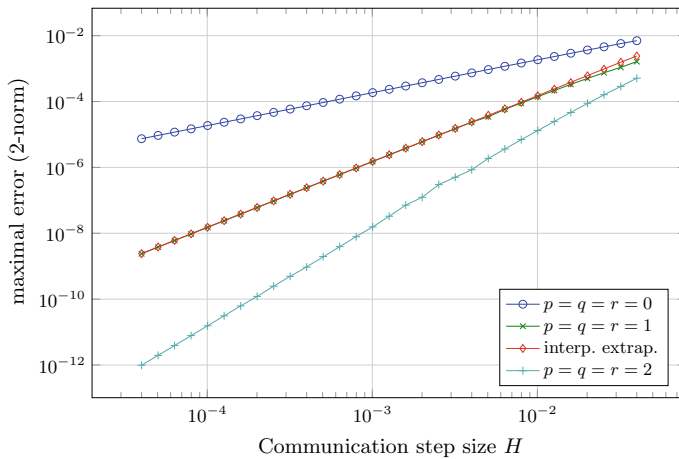


Fig. 6.6 Error plot for Test 1 (basic algorithm)

10^{-8} in the master system and 10^{-9} in the slave system. We have used a reference solution that was obtained by solving the monolithic system with MATLAB’s Adams-Bashforth-Moulton PECE solver `ode113` of order up to 13 with very tight absolute and relative tolerances of 2.2×10^{-14} .

Test 1 In this test, we compared the results of the basic algorithm with the smooth road profile for equal inter- and extrapolation order $p = q$ of different magnitude. We

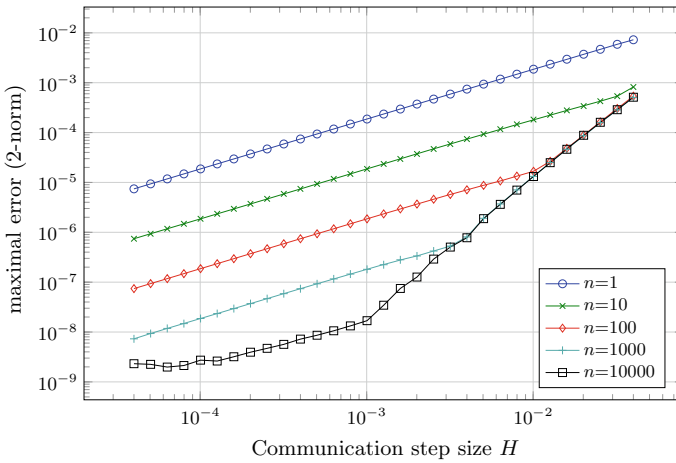


Fig. 6.7 Error plot for Test 2 (extended algorithm)

also assumed that the FMU is able to take inputs of sufficient order, thus $p = q = r$. Furthermore, we have also compared this to the case of linear interpolated extrapolation [5] and $p = r = 2$. The integration was done over $t \in [0, 4]$. The maximum of the errors in the 2-norm are plotted over the communication step size in Fig. 6.6. For polynomial extrapolation, we observe numerical convergence of order $p + 1$ for $p = q = r$. For interpolated extrapolation, we only get second order. This is consistent with the theoretical investigations in Sect. 6.5.

Test 2 Here, we want to show the effect of the number of substeps on the accuracy in the extended algorithm. We used polynomial inter- and extrapolation of second order $p = q = 2$ and assumed that the FMU only accepts piecewise constant inputs $r = 0$. Using the smooth road profile and the same tolerances and integration time span as above, we compared the results for varying amounts of substeps $n = 1, 10, \dots, 10000$. The resulting errors are, in the same fashion as before, depicted in Fig. 6.7.

The theory only gives us convergence of order $\mathcal{O}(H^3) + \mathcal{O}(H_S)$ and we can numerically observe the first order term in all cases for sufficiently small communication step sizes H . On the other hand we see, that an increase in the number of substeps decreases the term $\mathcal{O}(H_S)$ significantly. If the number of substeps is sufficiently large in relation to the communication step size, the overall result behaves as if the slave system would be able to take nonconstant inputs. What we see is that the resulting error is, for coarser communication step sizes bounded from below by the error we would get for $r > 0$, which is at most of third order, because $p = q = 2$.

Test 3 In this test, we wanted to check how the co-simulation algorithm behaves, when there are discontinuities in one of the models. Note that the theoretical investigations act on the assumption that all occurring functions are sufficiently smooth. We have used the same tolerances for solving the master and slave subsystems as above, but now the discontinuous step road profile was chosen and the integration time span was

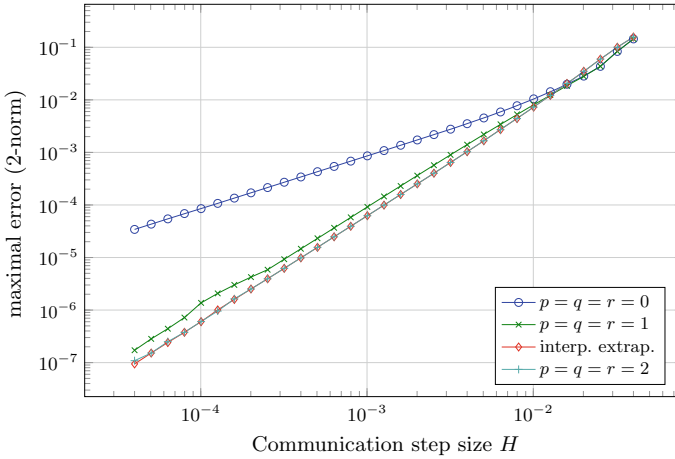


Fig. 6.8 Error plot for test 3

$t \in [0, 6]$. As in Test 1, we compared the results for $p = q = r$ for different values, and included interpolated extrapolation as well. Furthermore, we used the extended algorithm with $n = 10$ substeps for the test. The results are depicted in Fig. 6.8. We can see that the best convergence behaviour that can be numerically observed is quadratic. For $p = q = r = 0$, as before, only first order can be reached. Higher order inter- and extrapolation allows only for a slightly smaller error constant. Here, the interpolated extrapolation performs similarly as the polynomial extrapolation. This is mainly due to the fact that both algorithms use polynomial interpolation of second order to calculate the inputs for the master subsystem.

Verification with industrial tool We have verified these results by implementing a 3D version of the quarter car model with the industrial multi-body system tool Simpack. In Simpack, we created a mass and used a “user-force-element” in order to apply user-defined forces to the mass. The user force element is a Fortran code that can be programmed by the Simpack user. We used it to implement a Simpack API for the SNIWoWrapper and were able to simulate the system, where the SNIWoWrapper conducted the co-simulation of the slave system inside the FMU. We obtained similar results to the implementation above.

6.6.2 Industrial Application: Proof of Concept

The SNIWoWrapper was designed as testbed for numerical algorithms in academic as well as industrial co-simulation scenarios. We have used it in a proof-of-concept application as a way to include an FMU into a simulation tool.

During the SNIWoRed project we worked with a Simulink model of a front axle from an industrial partner. The model did not include a proper tire model, however.

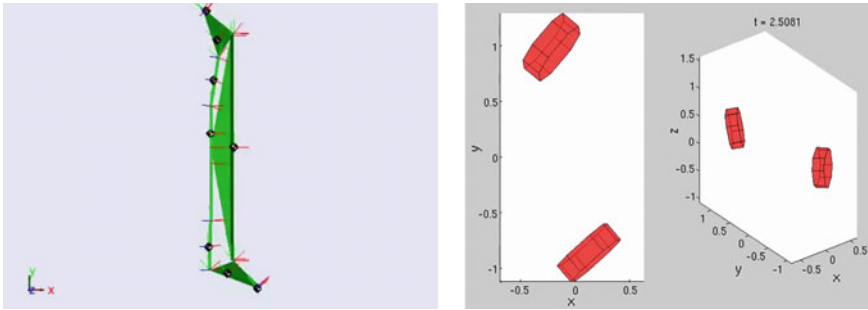


Fig. 6.9 Screenshots of the visual representation of the front axle model in Simulink (left) and of two of the commercial wheel models (right)

We wanted to combine a commercial tire model with the front axle model. In order to accomplish this, we have programmed an FMU that includes the commercial tire model by using its API. We have used the custom Simulink block which loads the SNiMoWrapper as shared library, see Sect. 6.4. With this configuration, we were able to perform a steering capacity test, where real-life data for the steering maneuver was used. The test consisted of fully steering to one side and then fully steering to the other side. The tire model worked properly and we were able to obtain meaningful data from this numerical experiment. Screenshots of the test are depicted in Fig. 6.9. Note that here we used simultaneously two instances of the same tire model FMU.

6.7 Conclusions

We have implemented an FMI v2.0 compatible tool that allows for easy incorporation of virtually any co-simulation FMU in standard simulation tools of nonlinear system dynamics. The co-simulation algorithm is contained entirely inside the SNiMoWrapper and thus hides the details of the co-simulation from the master simulation tool, essentially transforming the FMU into a function of time. For this to work, we only require mild prerequisites, such as that the integrator of the master simulation tool must step on each communication point before advancing to the next communication step. The co-simulation algorithm is of higher order and has some parameters which can easily be fine-tuned. The algorithm is of Gauss-Seidel type and incorporates the idea of substepping in order to deal with FMUs, that do not allow higher order signal extrapolation, and the idea of filtering the outputs of the slave system in order to deal with slave systems that are highly oscillating. We have given a proof of convergence that extends the proof given in [1].

Moreover, we have shown that the analytically predicted order of the algorithm can be observed numerically as well. For this, we have used the academic example of the quarter car model. Furthermore, we have shown an industrial proof-of-concept

example that goes beyond the world of academic examples. This shows that the SNiMoWrapper is applicable for real-life industrial problems as well.

Acknowledgements This work was supported by BMBF grant 05M10NHA [SNiMoRed] “Multidisciplinary simulation, nonlinear model reduction and pro-active control in vehicle dynamics.” The authors are grateful to the MDF team at Fraunhofer ITWM for providing the industrial test case “steering capacity” (Sect. 6.6.2) and for fruitful discussions on the SNiMoWrapper.

References

1. Arnold, M., Clauß, C., Schierz, T.: Error analysis and error estimates for co-simulation in FMI for model exchange and co-simulation v2.0. *Arch. Mech. Eng.* **LX**, 75–94 (2013). <https://doi.org/10.2478/meceng-2013-0005>
2. Arnold, M., Günther, M.: Preconditioned dynamic iteration for coupled differential-algebraic systems. *BIT* **41**, 1–25 (2001). <https://doi.org/10.1023/A:1021909032551>
3. Arnold, M., Hante, S., Köbis, M.: Error analysis for co-simulation with force-displacement coupling. *Proc. Angew. Math. Mech.* **14**, 43–44 (2014). <https://doi.org/10.1002/pamm.201410014>
4. Busch, M.: Continuous approximation techniques for co-simulation methods: analysis of numerical stability and local error. *ZAMM - J. Appl. Math. Mech./Zeitschrift für Angewandte Mathematik und Mechanik* **96**(9), 1061–1081 (2016). <https://doi.org/10.1002/zamm.201500196>
5. Dronka, S., Rauh, J.: Co-simulation-interface for user-force-elements. In: *Proceedings of SIM-PACK User Meeting, Baden-Baden* (2006)
6. FMI: The Functional Mock-up Interface. <http://fmi-standard.org/>
7. Hairer, E., Nørsett, S.P., Wanner, G.: *Solving Ordinary Differential Equations I, Nonstiff Problems*, 2nd edn. Springer, Berlin (2008). <https://doi.org/10.1007/978-3-540-78862-1>
8. Hairer, E., Wanner, G.: *Solving Ordinary Differential Equations II, Stiff and Differential-Algebraic Problems*, 2nd edn. Springer, Berlin (1996). <https://doi.org/10.1007/978-3-642-05221-7>
9. Kübler, R., Schiehlen, W.: Two methods of simulator coupling. *Math. Comput. Model. Dyn.* **6**, 93–113 (2000). [https://doi.org/10.1076/1387-3954\(200006\)6:2;1-M;FT093](https://doi.org/10.1076/1387-3954(200006)6:2;1-M;FT093)
10. Mitschke, M., Wallentowitz, H.: *Dynamik der Kraftfahrzeuge*, 4th edn. Springer, Berlin (2004). <https://doi.org/10.1007/978-3-662-06802-1>
11. Petzold, L.R.: A Description of DASSL: a differential/algebraic system solver. In: *Proceedings of IMACS World Congress* (1982)
12. Popp, K., Schiehlen, W.: *Ground Vehicle Dynamics*. Springer, Berlin (2010). <https://doi.org/10.1007/978-3-540-68553-1>
13. Walter, W.: *Ordinary Differential Equations*. Springer, Berlin (1998). <https://doi.org/10.1007/978-1-4612-0601-9>

Chapter 7

A Coupled Finite Element Analysis Approach Combining In-House and General-Purpose Codes



Tomoyoshi Horie, Tomoya Niho and Daisuke Ishihara

Abstract Coupled finite element analysis is expected in several fields of research and development, where necessity of coupled analysis of not only two phenomena but also three or more phenomena is increasing. The demand for multiscale and multiphysics analyses is also increasing. A coupled analysis approach that combines an in-house code and a general-purpose commercial finite element analysis code using a message passing interface is very promising for these analyses. To examine the practical usefulness and effectiveness, this approach was applied to the triply coupled and multiscale analyses of a resistance spot welding problem and to the dynamic coupled analysis of an ultra sonic piezoelectric motor. The approach is effective to perform various types of coupled finite element analyses of multiphysics and multiscale problems.

7.1 Introduction

Coupled finite element analysis is expected to become widespread in several fields of research and development, where there is a need to analyze several types of interactions among structural, fluid, thermal, current, and electromagnetic phenomena. Necessity of coupled analysis of not only two phenomena but also three or more phenomena has been increasing. Necessity of multiscale analysis as well as multiphysics analysis is also increasing.

There are several ways to analyze coupled problems. It is simple and straightforward to use a general-purpose finite element code having a capability for coupled analysis, but coupled phenomena that can be solved using such codes are limited. Another way is to develop a new analysis code containing specific coupled analysis

T. Horie (✉) · T. Niho · D. Ishihara
Kyushu Institute of Technology, 680-4 Kawazu, Iizuka, Fukuoka 820-8502, Japan
e-mail: horie@mse.kyutech.ac.jp, horie.tomoyoshi642@mail.kyutech.jp
URL: <http://www-solid.mse.kyutech.ac.jp/en/>

© Springer Nature Switzerland AG 2019
B. Schweizer (ed.), *IUTAM Symposium on Solver-Coupling and Co-Simulation*,
IUTAM Bookseries 35, https://doi.org/10.1007/978-3-030-14883-6_7

functions. Although this method can be applied to various types of coupled problems and coupled algorithms, the cost of development and verification will be a matter of concern. Then, the most practical way is to develop a new coupled analysis code combining two different existing analysis codes, but careful attention should be focussed on the fact that each code has its own time step loop. There are several approaches to combine different analysis codes, such as (i) merging the corresponding part of the time loops to build a unified analysis code, (ii) using data files, (iii) using coupling interface [1], and (iv) using message passing interface [2], for the purpose of exchanging the coupling terms of both codes. The final approach is considered to be quite simple and has a low development cost, in addition to being reliable and practical. From the view point of reliability, a general-purpose finite element analysis code is worth of utilization as one of the different analysis codes for the coupled analysis. The problem, however, is that the source code is not available, while it is also difficult to insert the time loop of an in-house code inside the time loop of a general-purpose code as a user subroutine.

We have proposed a coupled analysis approach by combining the in-house and general-purpose commercial FEM codes using message passing interface, and presented the specific procedures in this approach [3]. In this paper, the approach is applied to the triply coupled analysis of a resistance spot welding problem, the multiscale analysis of the welding problem, and the dynamic coupled analysis of an ultrasonic piezoelectric motor to examine the practical usefulness and effectiveness of this approach.

7.2 Coupled Analysis Approach

7.2.1 System Requirements

The requirements of coupled analysis using the proposed approach are as follows: (i) a user-developed in-house analysis code that contains special features or analysis functions suitable for a specific problem, (ii) a general-purpose FEM code containing a function of user subroutines, and (iii) a message passing interface that conforms to MPI-2 [2]. The in-house code should contain an original or unique algorithm that has not been realized in general-purpose FEM code, and the source code should be available for further modification. The general-purpose FEM code is reliable in general and consists of many functions, which is worth utilization as a structural analysis code although its source code is not available.

7.2.2 Concept of Approach

The analysis flow based on this approach is shown in Fig. 7.1. The in-house code is started first, then the general-purpose FEM code is initiated from the in-house code. Both codes run separately as different processes followed by the MPI initialization and the creation of a communicator for coupled terms before performing the time integration loop. Then, the coupling terms are exchanged using interprocess communication before the nonlinear iteration loop during each time step in the partitioned coupling algorithm. In the general-purpose FEM code, the MPI functions for coupled analysis are called only in the user subroutines.

7.2.3 Remarks in Implementation

Special attentions should be paid to the selection of user subroutines, the starting method of general-purpose code, coexistence with parallel processing, and compatibility of the MPI software.

Selection of user subroutines Since the general-purpose FEM code supports many types of user subroutines, it is critical to select the appropriate subroutines for the coupled algorithm, corresponding to the specific problem. The selection should be made according to the position where the subroutines are called in the algorithms such as the time and iteration loops. Further, attention should be paid to the arguments for physical quantities in each subroutine, such as the displacement, temperature, and reaction force. These are specified in general-purpose codes as nodal, element, or integration point variables. These quantities are also specified as input or output variables.

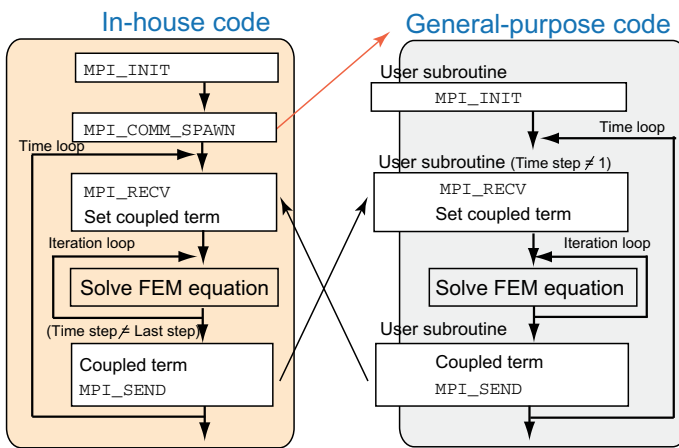


Fig. 7.1 Analysis flow using present approach

Starting general-purpose code Most of the general-purpose FEM codes are started by shell scripts; however, in-house codes have to start the general-purpose FEM code not by using shell script but by specifying the executable file with appropriate command-line arguments. This is because MPI-2 does not assure the successful execution of a shell script.

Coexistence with parallel processing Most of the general-purpose FEM codes are capable of parallel processing with the algorithm of the domain decomposition method using MPI. To realize the data communication between the in-house and general-purpose codes, another communicator, which is different from that used in parallel processing, should be used for the coupled analysis. Therefore, the in-house code should communicate not only with Rank 0 of the parallel general-purpose code that is corresponding to the spawned child but also with the remaining Ranks of parallel processes, as required. Sufficient attention should be devoted to avoid the double issue of `MPI_Init` in parallelized general-purpose FEM codes.

Compatibility of MPI software The product and version of MPI should be unified in the MPI communication. Since most of the general-purpose FEM codes contain a licensed version of MPI, the user should prepare the same MPI product to be used with the in-house code.

7.3 Applications to Coupled Analysis

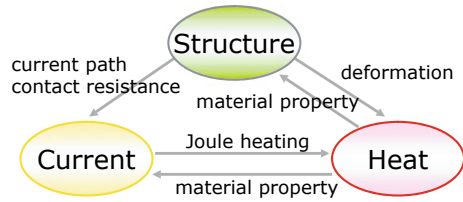
The coupled analysis approach was applied to develop a triply coupled code and a multiscale analysis code for a resistance spot welding problem. This approach was also applied to a dynamic coupled analysis code for an ultra sonic piezoelectric motor.

7.3.1 *Triply Coupled Analysis of Resistance Spot Welding*

Resistance spot welding is an extensively used bonding technique in industries because of its short welding time and minimal heat affect area near the weld zone. Although the welding process itself is quite simple, the phenomena during spot welding are complicated due to the triply coupled effect among the mechanical contact, current, and thermal conduction.

Coupled phenomena in resistance spot welding As illustrated in Fig. 7.2, the deformation and contact of steel sheets or structures affect the current and heat that are conducted in the structure. At the same time, the current and heat affect each other due to the Joule heating and the temperature dependence of material properties. The heat further affects the structures due to the temperature dependence of material properties. The contact resistance between the sheets depends strongly on both pressure and temperature.

Fig. 7.2 Triply coupled phenomena in resistance spot welding



Flow of triply coupled analysis The flow of triply coupled analysis is shown in Fig. 7.3. In current analysis, the temperature dependent material properties, pressure, and the temperature dependent contact resistance are taken into account. In transient thermal analysis, thermal contact resistance and Joule heat generation are also taken into account. Since these two analyses are strongly coupled, an in-house partitioned coupled code was developed with iterations between current and temperature [4]. With regard to structural analysis, a general-purpose structural analysis code MSC.Marc [5] is used to take into account the elasto-plasticity, contact, large displacement, finite strain, and temperature dependent material properties. Another partitioned method is applied for exchanging the temperature and contact properties between the in-house electric-thermal coupled analysis and general-purpose structural analysis codes using MPI.

Results of coupled Analysis The current and temperature distributions along with the nugget areas are shown in Fig. 7.4. In the early stage (2.0 ms), contact resistance heat generation appears at the center and edge of the sheet interface. In the middle

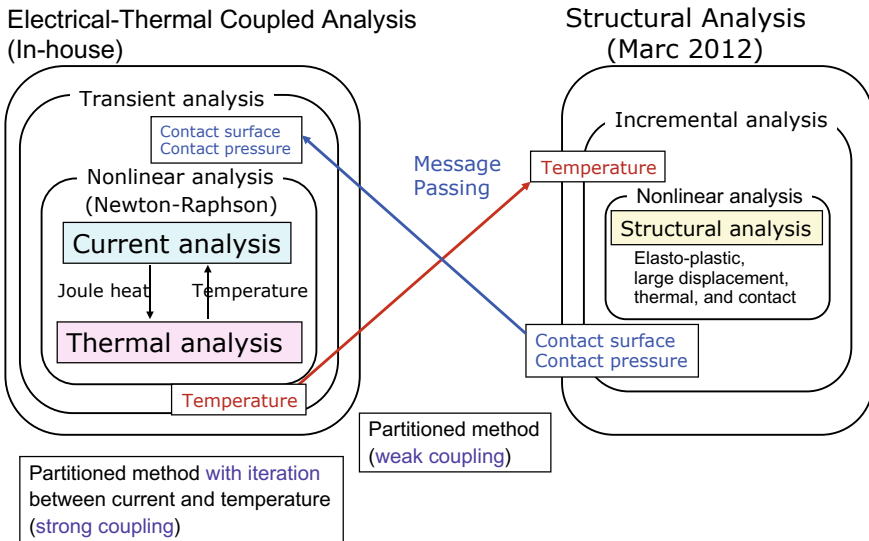


Fig. 7.3 The flow of triply coupled analysis of resistance spot welding

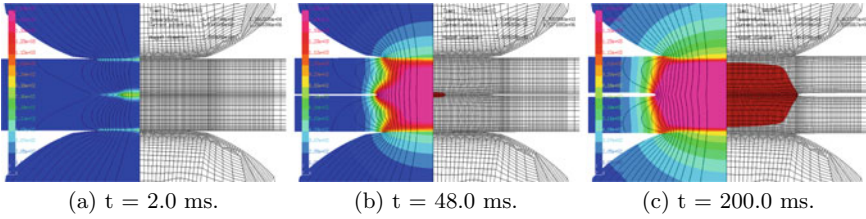


Fig. 7.4 Current and temperature distributions with the nugget area

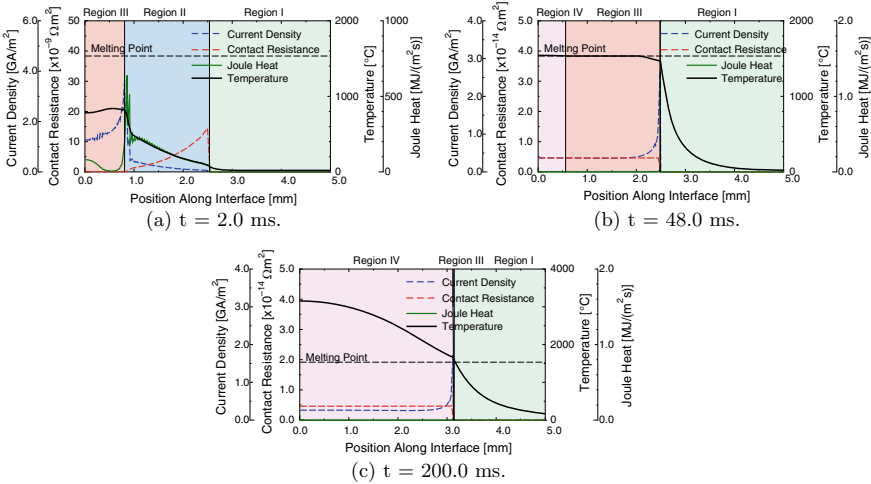


Fig. 7.5 The distributions of contact resistance, current density, Joule heat, and temperature

stage (48.0 ms), melting appears on the interface, then contact and nugget areas grow while the heat source is shifted from contact resistance heat generation to base metal heat generation. In the final stage, the nugget area approaches the contact edge.

Movement of the peaks in current density, contact resistance, and contact resistance heat generation along the sheet interface are shown in Fig. 7.5. The current density peak appears at the central axis in the early stage. In the middle stage, the current density and Joule heat peaks move outward. Then, the nugget is growing with the migration of the current density peak, Joule heat peak, contact resistance peak, and contact edge.

By combining the in-house and general-purpose FEM codes, the detailed examination of coupled phenomena that occur on the sheet interface in resistance spot welding is realized.

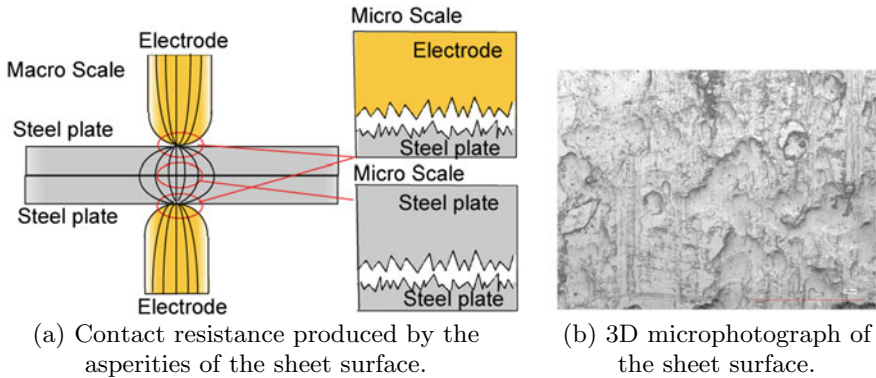


Fig. 7.6 Electrical contact resistance produced by the asperities between the sheet surfaces

7.3.2 Multiscale Analysis of Resistance Spot Welding

Microscopic model of the contact surface containing asperities is introduced to replace the contact resistance model based on both theory and experimental data.

Electrical contact resistance In resistance spot welding, electrical contact resistance exists on the interface of the sheets as seen in Fig. 7.6. It depends on the material properties of the sheet, asperity properties of the sheet surface (see Fig. 7.6b), contact pressure, and temperature. In the previous section, the contact resistance was modeled by an equation, which is a function of the contact pressure and temperature for each material, based on theory and experimental data [6]. The contact resistance, however, is not well understood theoretically, and it is difficult to perform the experiments at high temperatures. Therefore, a multiscale coupled analysis is proposed for the analysis of resistance spot welding. Further, the irreversibility of asperity deformation may be considered using multiscale analysis without conducting any additional experiments.

Multiscale coupled analysis of resistance spot welding Two scales are considered to perform resistance spot welding analysis as shown in Fig. 7.7. In macroscale analysis, a triply coupled mechanical contact, current, and thermal conduction analysis is performed in a similar manner as described in the previous section except for using the electrical contact resistance obtained by microscale analyses, where the contact analyses of small surfaces with asperities are performed using a general-purpose FEM code to obtain the electrical contact resistance. The contact pressure and temperature of each element surface obtained using macroscale analysis are further used in microscale analysis to obtain the electrical contact resistance of each macroscopic interface.

The microscale FE model of a statically similar representative volume element is shown in Fig. 7.8. The top surface configuration is modeled based on the data that

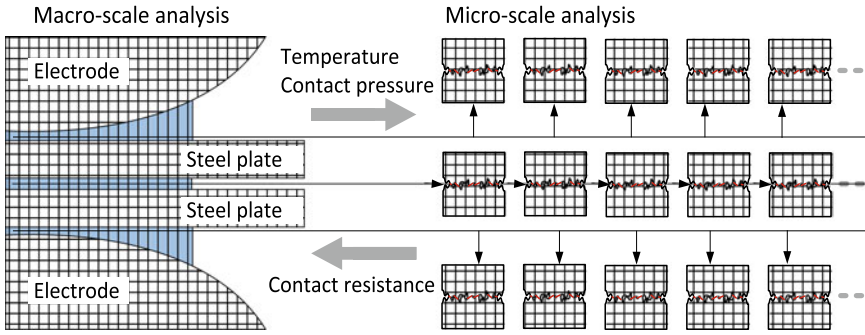


Fig. 7.7 Multiscale coupled analysis in which macroscale resistance spot welding analysis and microscale electrical resistance analyses are performed

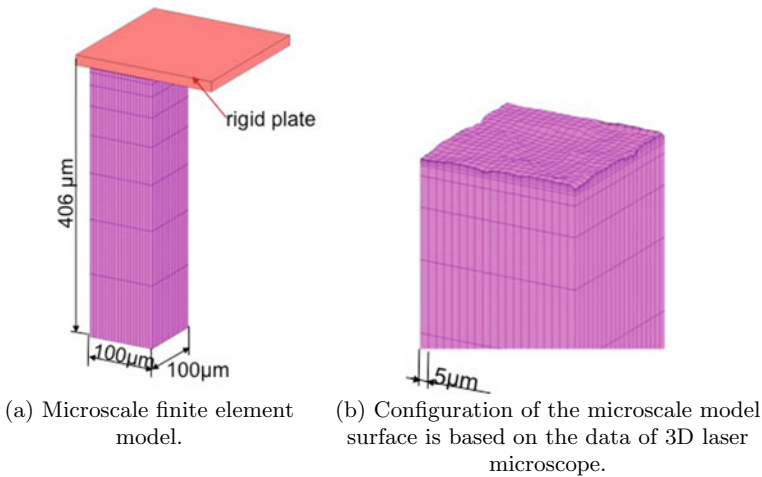


Fig. 7.8 Microscale analysis model of a statistically similar representative volume element

were obtained using a 3D laser microscope. The periodic boundary conditions are applied to the side surfaces. The element size of top surface is 5.0 μm in this model.

The flow of multiscale analysis is shown in Fig. 7.9. The triply coupled macroscale analysis code is started first, then the microscale code is initiated from the macroscale code. Using the temperature and pressure transferred from the macroscale analysis, the electrical contact resistance is obtained in the microscale analyses, then transferred to the macroscale analysis.

Results of multiscale coupled analysis The execution status of the microscale analyses on the sheet interface is shown in Fig. 7.10. When a new contact surface occurred, a new process to perform microscale analysis was created by MPI_Spawn. When the contact surface melted, the process terminated. Therefore, the multiscale analysis was realized using 40 microscale analyses, whose number is smaller than that of the

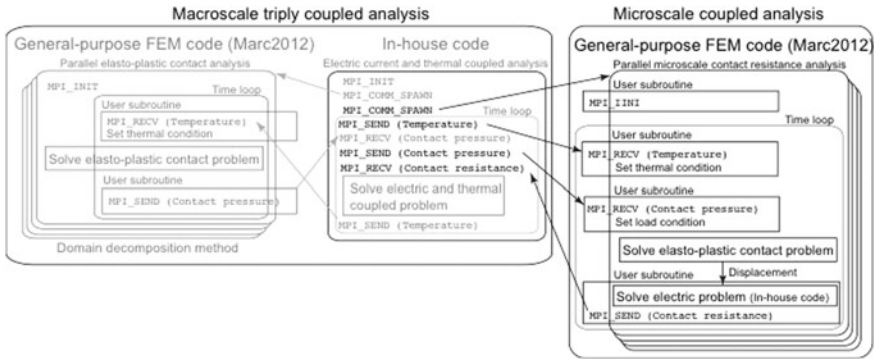


Fig. 7.9 Flow of the multiscale coupled analysis of resistance spot welding

Fig. 7.10 Execution status of microscale analyses during multiscale analysis

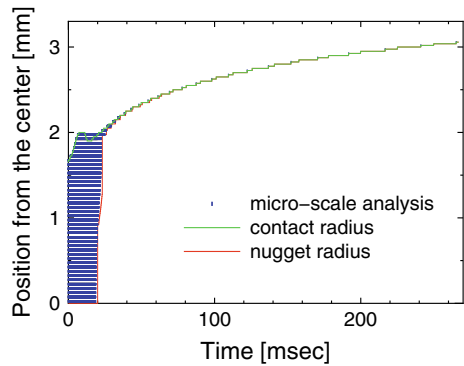
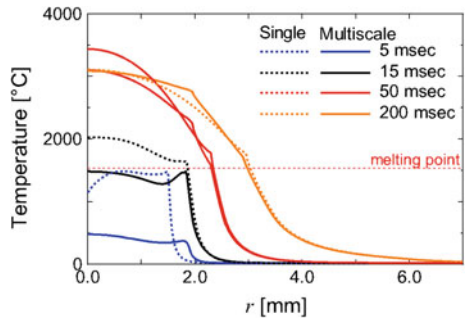


Fig. 7.11 Comparison of the temperature distribution along the interface for single scale and multiscale analyses



total contacted elements on the interface. The temperature distributions along the contacted surface are shown in Fig. 7.11. The results of the multiscale analysis are slightly lower than those of single scale finite element analysis. The nugget and contact surface growth curves are shown in Fig. 7.12. Although the nugget generation of multiscale analysis is delayed, the final diameter is approximately identical. These differences may arise from the resolution of asperity modeling.

Fig. 7.12 Comparison of the nugget and contact diameters for single scale and multiscale analyses

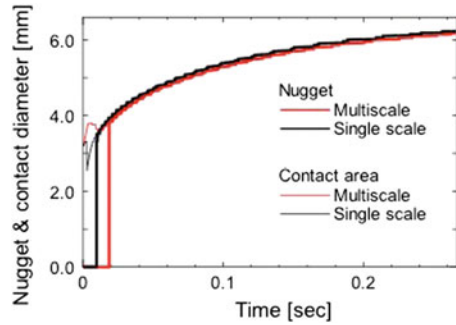
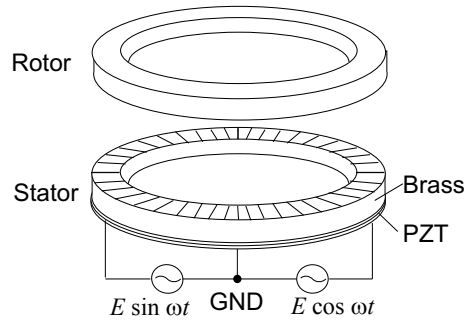


Fig. 7.13 Piezoelectric resonance driving ultrasonic motor



7.3.3 Coupled Analysis of Piezoelectric Ultrasonic Motor

Piezoelectric ultrasonic motors are used in small electronic devices such as an auto focus system of a camera because of their simple structure and quick response. The proposed approach was applied to an electric field and structural coupled analysis of the ultrasonic motor.

Driving mechanisms in piezoelectric ultrasonic motor As shown in Fig. 7.13, the ultrasonic motor consists of a rotor, a stator, and a high frequency electric power unit. The frequency of AC voltage is adjusted to generate traveling waves in the stator, which are excited due to inverse piezoelectric effect by piezoelectric elements aligned alternately in different polarization direction as shown in Fig. 7.14. Since the rotor is driven by the traveling waves of the stator through contact friction force [7], dynamic contact behavior is important for the analysis of ultrasonic motor. In the piezoelectric elements, direct piezoelectric effect as well as the inverse piezoelectric effect is induced as shown in Fig. 7.15. Therefore, the electric field and structural coupled analysis is also needed.

Coupled analysis of piezoelectric ultrasonic motor The flow of the electric field and structural coupled analysis using this approach is shown in Fig. 7.16. The electric field analysis with the direct piezoelectric effect for the stator is performed using an in-house code with the ϕ method, in which the equivalent nodal force due to the inverse

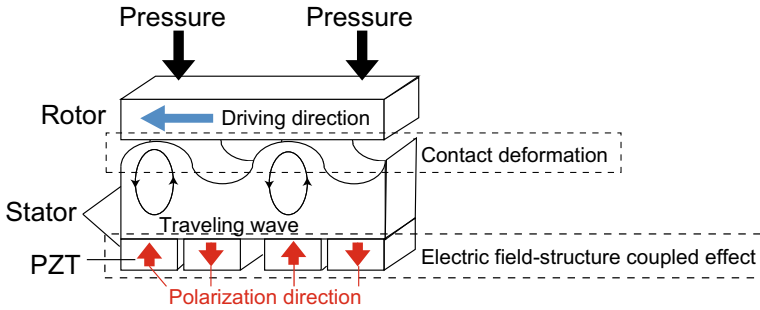


Fig. 7.14 Driving mechanism of the piezoelectric ultrasonic motor

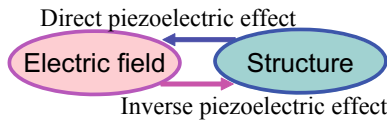


Fig. 7.15 Coupled effect in piezoelectric ultrasonic motor

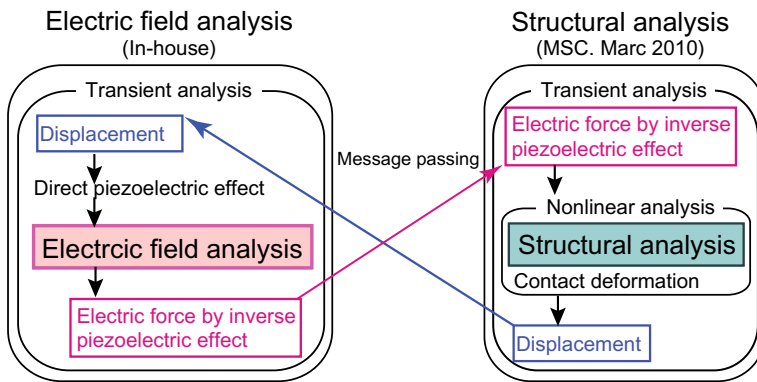


Fig. 7.16 The flow of coupled analysis of piezoelectric ultrasonic motor

piezoelectric effect is calculated at each time step. The dynamic structural analysis considering the equivalent nodal force and contact behavior between the rotor and stator is performed using a general-purpose structural analysis code MSC.Marc [5].

Results of coupled analysis of piezoelectric ultrasonic motor Conditions of the analysis are as follows: outer and inner diameters are 60 mm and 45 mm respectively, the thicknesses of the rotor and stator are both 2.5 mm, and the thickness of the piezoelectric elements is 0.5 mm. The applied voltage to the piezoelectric elements is 110 V with 40 kHz, while the rotor is pressed to the stator by 200 N force.

The results of the stator deformation and electric potential are shown in Fig. 7.17. Nine traveling waves that are similar to experiments are observed. Contact status

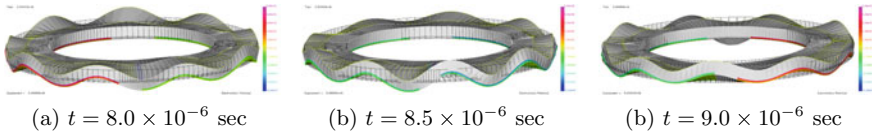


Fig. 7.17 Nine progressive waves appeared in the stator

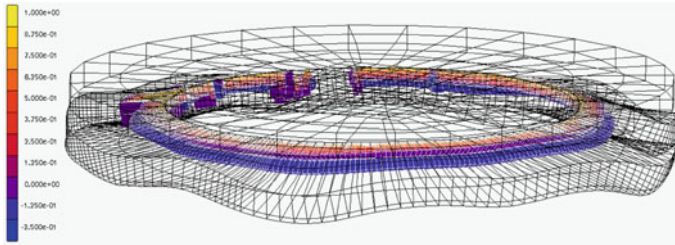


Fig. 7.18 Contact surface between stator and rotor

between the rotor and the stator is shown in Fig. 7.18. It is observed that the rotor is driven by the traveling waves of the stator.

The electric field and structural coupled analysis are realized by a general-purpose structural analysis code and an in-house code combined with the proposed approach. It is further confirmed that the proposed approach can be used for dynamic analysis.

7.4 Conclusions

A coupled analysis approach was proposed by combining an in-house code and a general-purpose FEM code using a message passing interface. The practical usefulness of this approach was examined by performing triply coupled analyses of resistance spot welding and ultra sonic piezoelectric motor, as well as by performing a multiscale analysis of resistance spot welding. The approach is effective for various types of coupled finite element analyses of both multiphysics and multiscale problems.

Acknowledgements This work was supported by JSPS KAKENHI Grant Number 16K05043. The authors wish to express their gratitude for the cooperation of TOYOTA MOTOR KYUSHU Inc.

References

1. Joppich, W., Kürschner, M.: Mpcci - a tool for the simulation of coupled applications. *Concurr. Comput. Pract. Exp.* **18**(2), 183–192 (2006). <https://doi.org/10.1002/cpe.913>
2. Message Passing Interface Forum: MPI: A Message–Passing Interface Standard Version 2.2. (2009). <http://mpi-forum.org/docs/>
3. Niho, T., Horie, T.: Coupled finite element analysis technique combining in-house code and commercial analysis code. *Trans. Jpn. Soc. Comput. Eng. Sci.* **2015**(20150015), (2015). <https://doi.org/10.11421/jsces.2015.20150015>
4. Niho, T., Horie, T., Uefuji, J., Ishihara, D.: Stability analysis and evaluation of staggered coupled analysis methods for electromagnetic and structural coupled finite element analysis. *Comput. Struct.* **178**, 129–142 (2017). <https://doi.org/10.1016/j.compstruc.2016.09.003>
5. MSC Software Corp.: Marc 2012 Volume A: Theory and User Information (2012). <https://simcompanion.mscsoftware.com/infocenter/index?page=content&id=DOC10193&>
6. Babu, S., Santella, M., Feng, Z., Riemer, B., Cohron, J.: Empirical model of effects of pressure and temperature on electrical contact resistance of metals. *Sci. Technol. Weld. Join.* **6**(3), 126–132 (2001). <https://doi.org/10.1179/136217101101538631>
7. Hagedorn, P., Wallaschek, J.: Travelling wave ultrasonic motors, part i: working principle and mathematical modelling of the stator. *J. Sound Vib.* **155**(1), 31–46 (1992). [https://doi.org/10.1016/0022-460X\(92\)90643-C](https://doi.org/10.1016/0022-460X(92)90643-C)

Chapter 8

Reduction of the Computation Time of Large Multibody Systems with Co-simulation Methods



Jan Kraft, Tobias Meyer and Bernhard Schweizer

Abstract Co-simulation methods can be used advantageously not only in the field of multidisciplinary simulations, but also to parallelize large monodisciplinary dynamical models. This paper focuses on the reduction of computation time that can be achieved in the simulation of multibody systems by partitioning a monolithic model into a variable number of coupled subsystems. The connection between the subsystems can be described in various ways. In this work, different subsystems are coupled by nonlinear constitutive equations (applied-force coupling approach). Exchange of coupling information takes only place at distinct macro-time points. The essential point is that the subsystems are integrated independently of each other between the macro-time points. If a Jacobi-type co-simulation scheme is used, all subsystems can be solved in parallel.

8.1 Introduction

Co-simulation or solver coupling methods are used in various fields of applications. Examples can be found in [1, 2]. The basic idea of co-simulation is to decompose an overall system into coupled subsystems. The formulation of the coupling conditions between two (or several) subsystems depends on the considered problem. In the case of mechanical systems, the decomposition of an overall model may be achieved by cutting through joints or by cutting through elements representing a physical force (torque). This leads to a coupling by reaction forces /torques

J. Kraft (✉) · T. Meyer · B. Schweizer
Technische Universität Darmstadt, Otto-Bernd-Str. 2, 64287 Darmstadt, Germany
e-mail: kraft@ad.tu-darmstadt.de

T. Meyer
e-mail: meyer@ad.tu-darmstadt.de

B. Schweizer
e-mail: schweizer@ad.tu-darmstadt.de

[3, 4] (constraint coupling) or to a connection by applied forces/torques (applied-force coupling). Co-simulation methods may further be subdivided into force/force-, force/displacement- and displacement/displacement-coupling approaches [5]. In this contribution, a force/force-decomposition approach is used and the subsystems are connected by nonlinear spring/damper-elements.

The methods presented here are weak coupling approaches, which implies that each subsystem is solved independently from the other subsystems within a macro-time step. Information (i.e. coupling variables) is only exchanged between the subsystems at certain communication-time points. The unknown coupling variables are approximated (extrapolated/interpolated) in the subsystems within a macro-time step. The independent integration of the subsystems within the macro-time steps is the essential point for parallelizing the computation.

In this manuscript, two different numerical methods for solving the coupled problem are examined: an explicit co-simulation technique and a semi-implicit integration scheme. The semi-implicit method is based on a predictor/corrector procedure, where the corrector step is carried out only once.

8.2 Co-simulation Methods

To investigate the performance of the explicit and the semi-implicit co-simulation approach with regard to the computation time, we use a nonlinear dynamical test model, which is described in the following subsection.

8.2.1 Nonlinear Test Model

One requirement for the test model is that it is straightforward to scale with respect to the number of degrees of freedom and with regard to the number of subsystems. Therefore, a chain of n_K masses connected by nonlinear spring/damper-elements is used as test model (Fig. 8.1).

Denoting the displacements of the oscillator-masses by the displacement coordinates x_i and the corresponding velocities by v_i , we obtain a system of $2n_K$ ordinary differential equations of the form

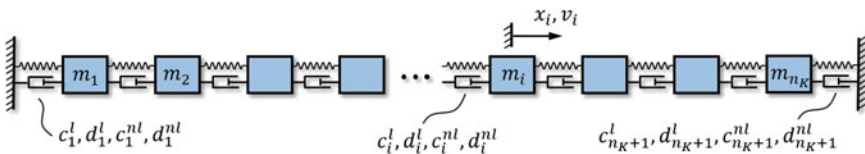


Fig. 8.1 Nonlinear test model: oscillator chain

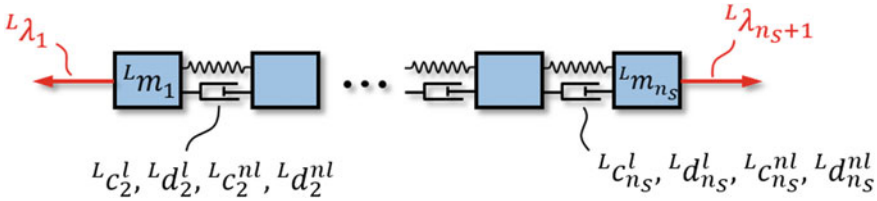


Fig. 8.2 Arbitrary subsystem L

$$\begin{aligned} \dot{x}_i &= v_i \\ \dot{v}_i &= \frac{c_i^l}{m_i}(x_{i-1} - x_i) + \frac{d_i^l}{m_i}(v_{i-1} - v_i) + \frac{c_{i+1}^l}{m_i}(x_{i+1} - x_i) + \frac{d_{i+1}^l}{m_i}(v_{i+1} - v_i) \\ &\quad + \frac{c_i^{nl}}{m_i}(x_{i-1} - x_i)^3 + \frac{d_i^{nl}}{m_i}(v_{i-1} - v_i)^3 + \frac{c_{i+1}^{nl}}{m_i}(x_{i+1} - x_i)^3 + \frac{d_{i+1}^{nl}}{m_i}(v_{i+1} - v_i)^3 \end{aligned}$$

with $i = 1 \dots n_K$. We assume that the chain is fixed at both ends ($x_0 = v_0 = x_{n_K+1} = v_{n_K+1} = 0$). The model parameters are the masses m_i , the linear stiffness coefficients c_i^l , the linear damping coefficients d_i^l , the nonlinear stiffness coefficients c_i^{nl} and the nonlinear damping coefficients d_i^{nl} .

8.2.2 Decomposition of the Test Model

As mentioned above, the overall system is split into coupled subsystems by a force/force-decomposition approach [5] (Fig. 8.2). This is achieved by cutting through certain nonlinear spring/damper-elements and by using the corresponding forces as coupling variables (Fig. 8.3). The number of subsystems n_{sub} is arbitrary, but usually much smaller than the number n_K of degrees of freedom of the overall system.

The set of n_s equations of motion for a subsystem L reads as

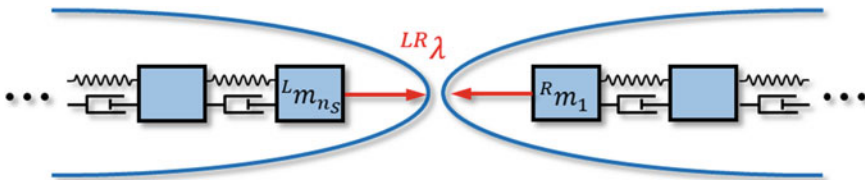


Fig. 8.3 Coupling of two adjacent subsystems L and R

$$\begin{aligned}
{}^L\dot{x}_j &= {}^L v_j \\
{}^L\dot{v}_j &= \frac{{}^L c_j^l}{L m_j} ({}^L x_{j-1} - {}^L x_j) + \frac{{}^L d_j^l}{L m_j} ({}^L v_{j-1} - {}^L v_j) + \frac{{}^L c_{j+1}^l}{L m_j} ({}^L x_{j+1} - {}^L x_j) \\
&\quad + \frac{{}^L d_{j+1}^l}{L m_j} ({}^L v_{j+1} - {}^L v_j) + \frac{{}^L c_j^{nl}}{L m_j} ({}^L x_{j-1} - {}^L x_j)^3 + \frac{{}^L d_j^{nl}}{L m_j} ({}^L v_{j-1} - {}^L v_j)^3 \\
&\quad + \frac{{}^L c_{j+1}^{nl}}{L m_j} ({}^L x_{j+1} - {}^L x_j)^3 + \frac{{}^L d_{j+1}^{nl}}{L m_j} ({}^L v_{j+1} - {}^L v_j)^3 - \frac{{}^L \lambda_j}{L m_j} + \frac{{}^L \lambda_{j+1}}{L m_j}
\end{aligned}$$

with $j = 1 \dots n_S$ and ${}^L c_1^l = {}^L d_1^l = {}^L c_1^{nl} = {}^L d_1^{nl} = {}^L c_{n_S+1}^l = {}^L d_{n_S+1}^l = {}^L c_{n_S+1}^{nl} = {}^L d_{n_S+1}^{nl} = 0$. The coupling forces are denoted by ${}^L \lambda_j$ and ${}^L \lambda_{j+1}$; they are only required for the coupling bodies ($j = 1$ and $j = n_S$) and are set to zero for the remaining bodies (${}^L \lambda_2 = \dots = {}^L \lambda_{n_S} = 0$).

The coupling condition for two adjacent subsystems L and R reads (assuming that body n_S of subsystem L is coupled with body 1 of subsystem R)

$$\begin{aligned}
{}^{LR}g &:= {}^{LR}\lambda - {}^{LR}c_c^l ({}^R x_1 - {}^L x_{n_S}) - {}^{LR}d_c^l ({}^R v_1 - {}^L v_{n_S}) \\
&\quad - {}^{LR}c_c^{nl} ({}^R x_1 - {}^L x_{n_S})^3 - {}^{LR}d_c^{nl} ({}^R v_1 - {}^L v_{n_S})^3 = 0
\end{aligned} \tag{8.1}$$

with the coupling force ${}^{LR}\lambda = {}^L \lambda_{n_S+1} = {}^R \lambda_1$, the coupling parameters ${}^{LR}c_c^l$, ${}^{LR}d_c^l$, ${}^{LR}c_c^{nl}$ and ${}^{LR}d_c^{nl}$, and the state variables of the two coupling bodies.

8.2.3 Explicit Co-simulation Scheme

To solve the decomposed system as a coupled problem by using an explicit co-simulation method, a macro-time grid is introduced. Within an arbitrary macro-step from T_N to $T_{N+1} = T_N + h_{mac}$, each subsystem is integrated using extrapolation (interpolation) polynomials of degree n_{pol} to approximate the coupling forces. After the integration of the subsystems, the resulting states of the coupling bodies are substituted into the coupling condition (8.1) to obtain the coupling force at the new macro-time point T_{N+1} (update of the coupling variables). The process of approximating and updating the coupling forces is presented schematically for an arbitrary coupling force λ ($= {}^{LR}\lambda$) in Fig. 8.4. Note that for the reason of a concise representation, the subsystem indices have been omitted. The approximation polynomial $\lambda^{(o)}(t)$ (the upper index (o) stands for ‘‘original’’ approximation polynomial) is defined by

$$\begin{aligned}
\lambda_{N+1}^{(o)} &= P_{n_{pol}} \left([T_{N-n_{pol}}, \lambda_{N-n_{pol}}], \dots, [T_N, \lambda_N], T_{N+1} \right) \\
\lambda^{(o)}(t) &= P_{n_{pol}} \left([T_{N-n_{pol}+1}, \lambda_{N-n_{pol}+1}], \dots, [T_{N+1}, \lambda_{N+1}^{(o)}], t \right).
\end{aligned} \tag{8.2}$$

Fig. 8.4 Explicit co-simulation: (linear) approximation polynomial $\lambda^{(o)}(t)$ for an arbitrary coupling force λ

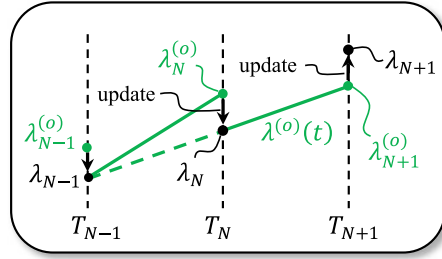
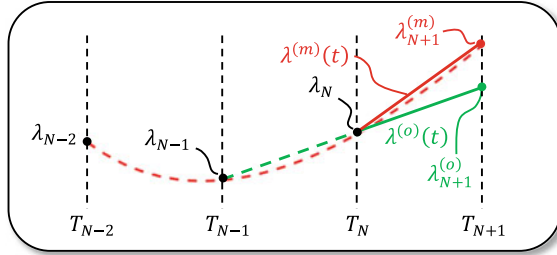


Fig. 8.5 Approximation polynomials (linear) of an arbitrary coupling force λ , green: original approximation polynomial, red: modified approximation polynomial for the error estimation



The abbreviation P_i represents inter-/extrapolation polynomials of order i defined by the sampling points at the macro-time points. The explicit co-simulation method has the advantage that a repetition of the macro-step is not necessarily required if a constant macro-step size is used (i.e. for the case that a macro-step size controller is not used). This may be an important point, if commercial subsystem solvers are used, which often do not allow solver reinitialization.

Macro-Step Size Controller. The explicit co-simulation method may be improved by using a macro-step size controller. To estimate the error of a macro-step, a second subsystem integration process within each macro-step is required. For the second integration, the sampling point $\lambda_{N+1}^{(o)}$ of the (original) approximation polynomial $\lambda^{(o)}(t)$ of the coupling force at the new macro-point T_{N+1} is modified. The modified value $\lambda_{N+1}^{(m)}$ of the approximated coupling force is obtained by increasing the extrapolation order by one ($n_{pol} + 1$). It has to be noted that the extrapolation order is only increased to obtain the modified value for the sampling point $\lambda_{N+1}^{(m)}$, the polynomial order of the modified approximation polynomial $\lambda^{(m)}(t)$ remains n_{pol} . This process is illustrated for an arbitrary coupling force λ in Fig. 8.5. To clarify the procedure of generating the approximation polynomial $\lambda^{(m)}(t)$ for the error estimator, the equations to obtain the sampling point and the corresponding approximation polynomial are presented in (8.3).

$$\begin{aligned} \lambda_{N+1}^{(m)} &= P_{n_{pol}+1}([T_{N-n_{pol}-1}, \lambda_{N-n_{pol}-1}], \dots, [T_N, \lambda_N], T_{N+1}) \\ \lambda^{(m)}(t) &= P_{n_{pol}}([T_{N-n_{pol}+1}, \lambda_{N-n_{pol}+1}], \dots, [T_{N+1}, \lambda_{N+1}^{(m)}], t) \end{aligned} \quad (8.3)$$

The resulting states of the two subsystem integration processes (with the original and the modified approximation polynomial of the coupling force) are used to estimate the error ε of the macro-step according to the following equation

$$\varepsilon = \sqrt{\sum_{\substack{\text{coupling} \\ \text{bodies}}} \left(\left(\frac{C_\varepsilon \cdot (x_{N+1,i}^{(o)} - x_{N+1,i}^{(m)})}{\text{atol}_i + \text{rtol} \cdot x_{N+1,i}^{(o)}} \right)^2 \right)}. \quad (8.4)$$

The error constant C_ε depends on the extrapolation order and the step sizes of the previous macro-steps. The position variables $x_{N+1,i}^{(o)}$ and $x_{N+1,i}^{(m)}$ are the displacements of body i at T_{N+1} obtained by the subsystem integration with the original and with the modified approximation polynomial of the coupling force. The error ε is the estimated local error on position level. An error estimator on velocity level can be constructed in the same way. The absolute and relative error tolerances atol_i and rtol are user defined values. The new macro-step size is determined according to

$$h_{mac}^{new} = 0.9 \cdot \varepsilon^{-\frac{1}{n_{pol}+3}} \cdot h_{mac}^{old}, \quad (8.5)$$

where $n_{pol} + 3$ is the convergence order of the co-simulation method on position level. A detailed description of this macro-step size controller can be found in [6].

8.2.4 Semi-implicit Co-simulation Scheme

A detailed description of the implemented semi-implicit co-simulation procedure can be found in [5]. The basics of the approach are only briefly explained next.

The macro-time grid is assumed to be equidistant (macro-step size $h_{mac} = \text{const.}$). As mentioned above, the presented semi-implicit co-simulation method is based on a predictor/corrector approach with only one corrector step. An arbitrary macro-time step from T_N to T_{N+1} is explained next for a co-simulation with two subsystems L and R (assuming that body n_S of subsystem L is coupled with body 1 of subsystem R).

Predictor Step. Within the predictor step, each subsystem is integrated twice from T_N to T_{N+1} : firstly with the predicted (extrapolated) coupling force $\lambda^P(t)$ ($= {}^{LR}\lambda^P = {}^L\lambda_{n_S+1}^P = {}^R\lambda_1^P$) and secondly with the perturbed predicted coupling force $\lambda^\Delta(t)$ ($= {}^{LR}\lambda^\Delta = {}^L\lambda_{n_S+1}^\Delta = {}^R\lambda_1^\Delta$). Note that for the reason of a concise representation, the subsystem indices have been omitted. The sampling point λ_{N+1}^Δ of the perturbed coupling force at T_{N+1} is obtained by adding a small, user-defined perturbation $\Delta\lambda$ to the sampling point λ_{N+1}^P of the predicted coupling force as given in (8.6).

$$\lambda_{N+1}^P = P_{n_{pol}} \left([T_{N-n_{pol}}, \lambda_{N-n_{pol}}], \dots, [T_N, \lambda_N], T_{N+1} \right)$$

$$\lambda_{N+1}^{\Delta} = \lambda_{N+1}^p + \Delta\lambda \quad (8.6)$$

$$\begin{aligned} \lambda^p(t) &= P_{n_{poi}} \left([T_{N-n_{poi}+1}, \lambda_{N-n_{poi}+1}], \dots, [T_{N+1}, \lambda_{N+1}^p], t \right) \\ \lambda^{\Delta}(t) &= P_{n_{poi}} \left([T_{N-n_{poi}+1}, \lambda_{N-n_{poi}+1}], \dots, [T_{N+1}, \lambda_{N+1}^{\Delta}], t \right) \end{aligned} \quad (8.7)$$

With the predicted state variables z^p and the perturbed predicted states z^{Δ} at T_{N+1} , the partial derivatives of the states with respect to the coupling variables can be approximated by finite differences

$$\left. \frac{\partial z_c}{\partial \lambda} \right|_{\lambda^p} = \lim_{\Delta\lambda \rightarrow 0} \frac{z_c(\lambda^p + \Delta\lambda) - z_c(\lambda^p)}{\Delta\lambda} \approx \frac{z_c^{\Delta} - z_c^p}{\Delta\lambda}. \quad (8.8)$$

Note that partial derivatives only have to be calculated for the coupling bodies.

Corrector Step. The approximated partial derivatives obtained in the predictor step are utilized to compute the improved (corrected) coupling force. Therefore, the coupling condition (8.1) is considered as a function of the coupling force λ at T_{N+1} and expanded in a Taylor series. Choosing λ_{N+1}^p as expansion point and neglecting higher-order terms $\mathcal{O}(\lambda^2)$, one obtains the linearized coupling condition

$$\begin{aligned} g^{lin}(\lambda) &:= g(\lambda^p) + \left. \frac{\partial g}{\partial \lambda} \right|_{\lambda^p} (\lambda - \lambda^p) \\ &= \lambda^p - c_c^l ({}^R x_1^p - L x_{n_s}^p) - d_c^l ({}^R v_1^p - L v_{n_s}^p) - c_c^{nl} ({}^R x_1^p - L x_{n_s}^p)^3 \\ &\quad - d_c^{nl} ({}^R v_1^p - L v_{n_s}^p)^3 \\ &\quad + \left[1 - c_c^l \left(\left. \frac{\partial {}^R x_1^p}{\partial \lambda} \right|_{\lambda^p} - \left. \frac{\partial L x_{n_s}^p}{\partial \lambda} \right|_{\lambda^p} \right) - d_c^l \left(\left. \frac{\partial {}^R v_1^p}{\partial \lambda} \right|_{\lambda^p} - \left. \frac{\partial L v_{n_s}^p}{\partial \lambda} \right|_{\lambda^p} \right) \right. \\ &\quad - 3c_c^{nl} ({}^R x_1^p - L x_{n_s}^p)^2 \left(\left. \frac{\partial {}^R x_1^p}{\partial \lambda} \right|_{\lambda^p} - \left. \frac{\partial L x_{n_s}^p}{\partial \lambda} \right|_{\lambda^p} \right) \\ &\quad \left. - 3d_c^{nl} ({}^R v_1^p - L v_{n_s}^p)^2 \left(\left. \frac{\partial {}^R v_1^p}{\partial \lambda} \right|_{\lambda^p} - \left. \frac{\partial L v_{n_s}^p}{\partial \lambda} \right|_{\lambda^p} \right) \right] (\lambda - \lambda^p) = 0. \end{aligned} \quad (8.9)$$

Solving Eq. (8.9) for the coupling force λ yields the corrected coupling force λ_{N+1}^c . In general, the predicted state variables and the predicted coupling force will not fulfill the coupling condition. The corrected coupling force (together with the corrected state variables), however, fulfills at least the linearized coupling condition (8.8). The subsystem integration within the corrector step is carried out by making use of the corrected coupling force $\lambda^c(t) = P_{n_{poi}} \left([T_{N-n_{poi}+1}, \lambda_{N-n_{poi}+1}], \dots, [T_{N+1}, \lambda_{N+1}^c], t \right)$.

The corrected state variables together with the corrected coupling forces will in general not fulfill the nonlinear coupling conditions. To achieve consistent coupling forces, an update of the coupling forces at T_{N+1} is useful. Therefore, the corrected state variables of the coupling bodies are substituted into the coupling condition (8.1) in order to calculate updated coupling forces.

8.2.5 Subsystem Solver

The subsystems are solved with the IDA solver from the SUNDIALS (Suite of Nonlinear and Differential/Algebraic Equation Solvers) package [7]. This implicit DAE solver is based on a variable-order variable-coefficient BDF implementation combined with either direct (sparse) or iterative methods for solving the linear system within the Newton iteration. For the present studies, the direct sparse linear solver (KLU [8]) is used.

8.3 Remarks on the Computation Time

8.3.1 Parallelized Computation

Within a macro-step, each subsystem is integrated independently. Exchange of information takes only place before or after the subsystem integration processes. Therefore, all subsystems can be solved in parallel. The parallelized implementation is realized with a hybrid MPI-openMP code: each subsystem is executed by a MPI rank. The different integration processes of each subsystem within a macro-time step—these are the additional integrations for the error estimation in case that a macro-step size controller is used and the subsystem integrations with the perturbed coupling variables for the semi-implicit approach—are carried out in openMP threads that are spawned within each MPI rank (Fig. 8.6). The simulations for this work have been carried out on a high performance computer (Lichtenberg High Performance Computer of the TU Darmstadt) so that all subsystem integrations could be fully parallelized.

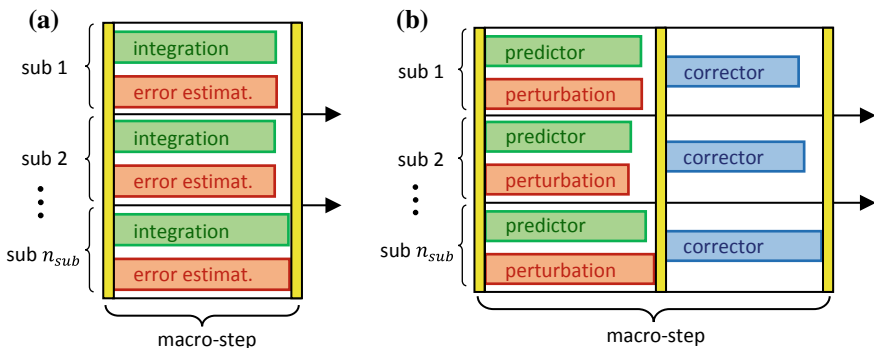


Fig. 8.6 Parallelization scheme: **a** explicit co-simulation (with macro-step size controller) and **b** semi-implicit co-simulation

Applying a parallel implementation, the simulation time is usually strongly reduced. The computation time for the co-simulation can be estimated by

$$T_{cos}^{(expl)} \approx \frac{T_{mon}}{n_{sub}^P} + C^{(expl)} \quad \text{or} \quad T_{cos}^{(semi)} \approx 2 \frac{T_{mon}}{n_{sub}^P} + C^{(semi)}, \quad (8.10)$$

where T_{mon} denotes the computation time of the monolithic model and n_{sub} the number of subsystems. P represents the scaling factor of the computation time of the multibody implementation with respect to the number of degrees of freedom. For typical multibody systems, the value of P is between one and three, depending on the formulation of the equations of motion and the solving strategy. The overhead caused by the synchronization of parallel threads and additional calculations due to the co-simulation approach (e.g. solving Eqs. (8.8) and (8.9) for the semi-implicit method) is summarized in the parameter C . The formula for T_{cos} implies the assumption that the overall system is split into equal-sized subsystems, so that the integration times for the different subsystems are similar.

8.3.2 Micro-step Size Limitation

Depending on the subsystem solver, there are different options to control the micro-step size (subsystem solver step size). The most obvious possibility is to enforce the subsystem solver to stop exactly at the end of each macro-step (“exact stop”, $t_{N+1}^{(sub)} = T_{N+1}$). The problem with this option is that the step-size controller of the subsystem solver will be interrupted. This may lead to a significant increase of the number of micro-steps. A completely unrestricted micro-step size on the other hand is also undesirable, because then it can happen that the micro-step size of a subsystem is larger than the macro-step size. As a result, the subsystem will not be evaluated in each macro-step.

Another possibility is to restrict the subsystem solver step size by the macro-step size ($h_{mic} \leq h_{mac}$) and to stop the solver when it steps beyond a macro-time point ($t_{N+1}^{(sub)} \geq T_{N+1}$). The states of the coupling bodies are interpolated at the macro-time point T_{N+1} . Within the next macro-step, the subsystem solver starts at the point t_{sub} at which it has stopped before. In this case, the subsystem solver step size controller does not get affected by the co-simulation. However, the interval between the end of a macro-step T_{N+1} and the point $t_{N+1}^{(sub)}$ at which the solver stops is integrated with the coupling force from the previous macro-step. This leads to an additional numerical error and an increased discontinuity in the coupling force. In our simulations, we observed that the restriction $h_{mic} \leq h_{mac}$ yields good results.

Figure 8.7 shows the micro-step size of an arbitrary subsystem solver of an explicit co-simulation carried out with the two different micro-step size limitations. As can be seen, the micro-step size of the co-simulation in which the subsystem solver is stopped exactly at each macro-time point (Fig. 8.7b) shows large fluctuations. The

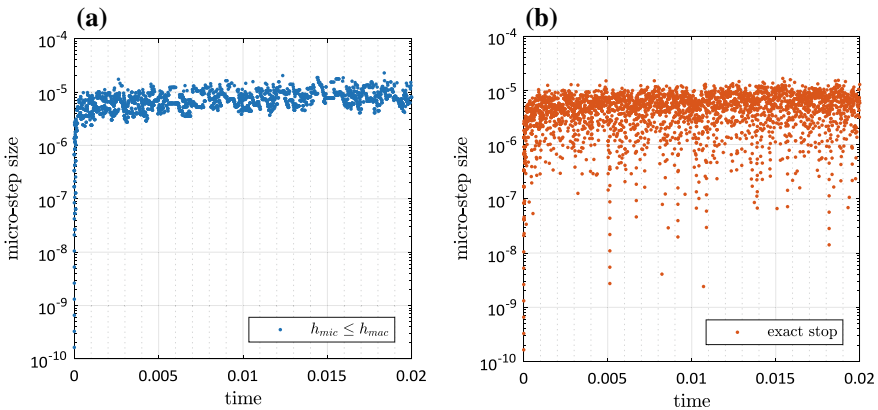


Fig. 8.7 Micro-step size limitation: **a** $h_{mic} \leq h_{mac}$ and **b** integration is stopped at each macro-time point exactly

average micro-step size is therefore smaller than for the co-simulation in which the micro-step size is restricted by $h_{mic} \leq h_{mac}$.

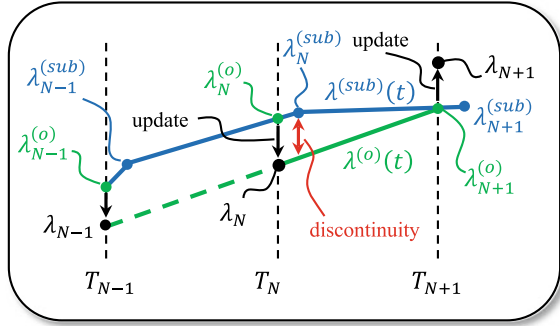
8.3.3 Macro-step Size

Considering classical time integration methods, the numerical error decreases and the computation time increases when the time step size is reduced. For co-simulation methods, the numerical error also decreases with the macro-step size, of course, but the relation of the computation time and the macro-step size is not necessarily as straightforward as for classical time integration methods.

Assuming that the co-simulation is carried out in parallel, the overall computation time is the sum of the subsystem integration time (i.e. of the slowest subsystem integration process) and the time required for synchronization and data exchange between the subsystems. The macro-step size can affect both parts of the overall computation time. On the one hand, a larger macro-step size decreases the number of synchronization points and the data transfer between the subsystems; therefore, it reduces the computation time. On the other hand, a larger macro-step size increases the discontinuities in the coupling variables at the macro-time points. Because of these discontinuities, the subsystem solver has to reduce the micro-step size at the beginning of each macro-step. As a result the overall computation time may increase. A small macro-step size will limit the micro-step size and therefore also increase the number of micro-steps and the overall computation time.

For the determination of an appropriate macro-step size, it is necessary to know what the dominating part of the overall computation time is. If the bottleneck is the data transfer—this may be the case when there is a large number of subsys-

Fig. 8.8 Explicit co-simulation: continuous (linear) approximation polynomial (blue curve)



tems or coupling variables—then the macro-step size should be chosen as large as possible. If the dominating factor of the computation time are the subsystem integration processes (which is mostly the case when multibody systems are coupled) then an appropriate macro-step size is a compromise between small discontinuities and minimal limitation of the micro-step size.

The discontinuities in the coupling variables at the end of each macro-time step, that are induced by the explicit co-simulation approach, can be eliminated or at least reduced by a modification of the approximation polynomials. An approach to obtain C^1 -continuity in the coupling variables by using cubic spline interpolation (“extrapol”) is described in [9]. However, the additional discontinuities that occur, if the subsystem solver does not stop at each macro-time point exactly, will remain.

A similar modification of the approximation polynomials (“modip”), that completely removes the discontinuities, is illustrated in Fig. 8.8 (for linear approximation). The adjusted approximation polynomial $\lambda^{(sub)}(t)$ for the macro-step from T_N to T_{N+1} , is obtained by substituting the sampling point $[T_N, \lambda_N]$ with $[t_N^{(sub)}, \lambda_N^{(sub)}]$ as represented in (8.12). $t_N^{(sub)}$ is the time at which the subsystem solver stopped the integration within the previous macro-time step and $\lambda_N^{(sub)}$ is the corresponding value of the coupling variable. It should be mentioned that $t_N^{(sub)}$ (and $\lambda_N^{(sub)}$) can be different for each subsystem.

$$\lambda_{N+1}^{(o)} = P_{n_{pol}}([T_{N-n_{pol}}, \lambda_{N-n_{pol}}], \dots, [T_N, \lambda_N], T_{N+1}) \tag{8.11}$$

$$\begin{aligned} \lambda^{(sub)}(t) &= P_1([t_N^{(sub)}, \lambda_N^{(sub)}], [T_{N+1}, \lambda_{N+1}^{(o)}], t) \quad (n_{pol} = 1) \\ \lambda^{(sub)}(t) &= P_{n_{pol}}([T_{N-n_{pol}+1}, \lambda_{N-n_{pol}+1}], \dots, [t_N^{(sub)}, \lambda_N^{(sub)}], [T_{N+1}, \lambda_{N+1}^{(o)}], t) \end{aligned} \tag{8.12}$$

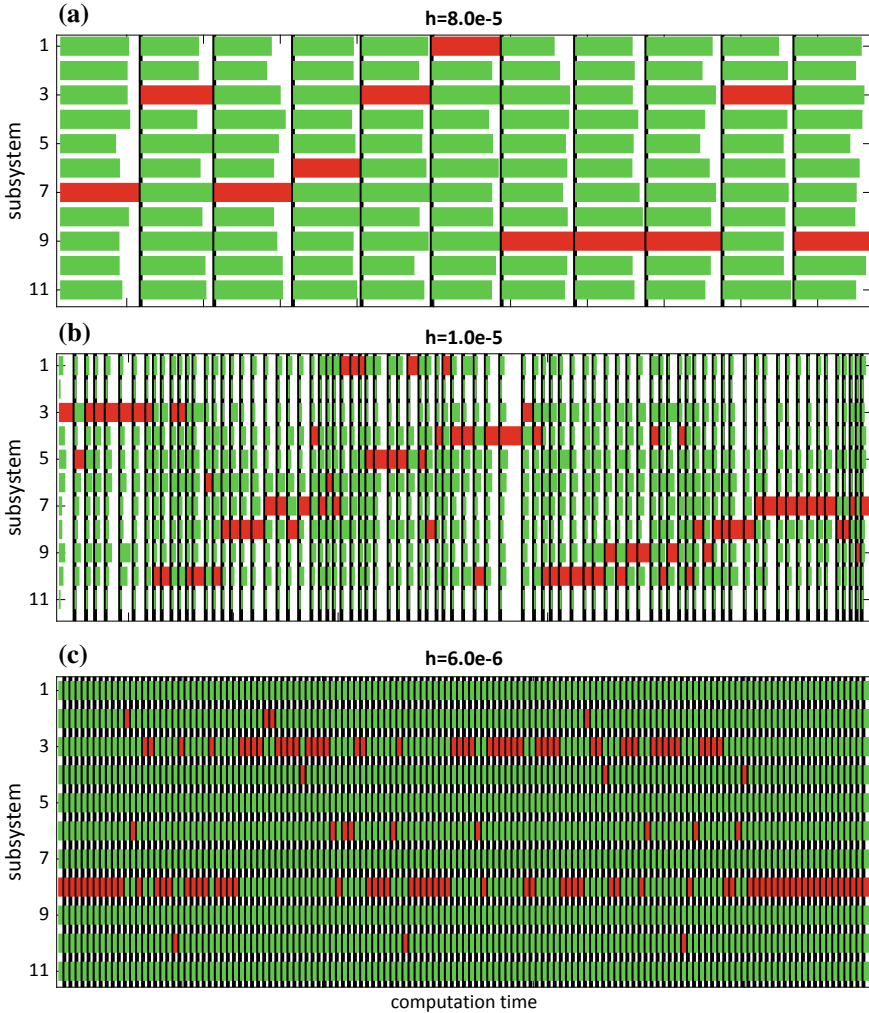


Fig. 8.9 Subsystem computation times for different macro-step sizes: **a** ~20–30 micro-steps per macro-step, **b** ~1–4 micro-steps per macro-step, **c** 1 micro-step per macro-step

8.3.4 Differences in Subsystem Computation Times

The computation time of each subsystem solver within each macro-step varies between the subsystems. Even if all subsystems have approximately the same computation time for the overall simulation, as in our test case, there are different computation times for each subsystem in each macro-step as shown in Fig. 8.9b. The effect of these “local” computation time fluctuations on the overall computation time depends strongly on the macro-step size.

Figure 8.9 shows a detailed view of the computation time of an explicit co-simulation with a constant macro-step size. The subsystems are physically identical. The green bars show the computation time of each subsystem in each macro-step. The macro-steps are indicated by the black lines. The red bars mark the slowest subsystem integration process in each macro-step.

In Fig. 8.9a the macro-step size is chosen relatively large so that each subsystem solver makes many micro-steps ($\sim 20\text{--}30$) within each macro-step. The computation times of the subsystems within each macro-step are similar.

Figure 8.9b shows results obtained with a reduced macro-step size. Here, the subsystem solver makes only a small number of micro-steps ($\sim 1\text{--}4$) within each macro-step. This results in relatively large differences of the subsystem computation times. Considering that each subsystem is assigned to one core, the hardware utilization is low, because the cores are idling a large proportion of the time.

The macro-step size in Fig. 8.9c is assumed to be smaller than the micro-step size that is chosen by the step size controller of the subsystem solver. Due to the restriction $h_{mic} \leq h_{mac}$, the micro-step size is limited by the macro-step size. Each subsystem solver takes only one micro-step per macro-step. As a result the computation time for all subsystems is almost equal. The price for the good hardware utilization is the increased number of micro-steps.

8.4 Simulation Results

To investigate the co-simulation methods the following parameter sets for the test model are defined in Table 8.1:

The external force F_{S_i} is defined by the force law

Table 8.1 Test model parameters

Parameter set 1 (“model 1”)	Parameter set 2 (“model 2”)
$n_K = 7750$	$n_K = 7750$
$n_{sub} = 47$	$n_{sub} = 47$
$m_i = 1.0e0$	$m_i = 1.0e0$
$c_i^l = c_{c_i}^l = 1.0e7$	$c_i^l = c_{c_i}^l = 1.0e7$
$d_i^l = d_{c_i}^l = 1.0e0$	$d_i^l = d_{c_i}^l = 1.0e0$
$c_i^{nl} = c_{c_i}^{nl} = 1.0e9$	$c_i^{nl} = c_{c_i}^{nl} = 1.0e9$
$d_i^{nl} = d_{c_i}^{nl} = 1.0e-2$	$d_i^{nl} = d_{c_i}^{nl} = 1.0e-2$
No external forces	External forces F_{S_i} acting on 5% of all bodies

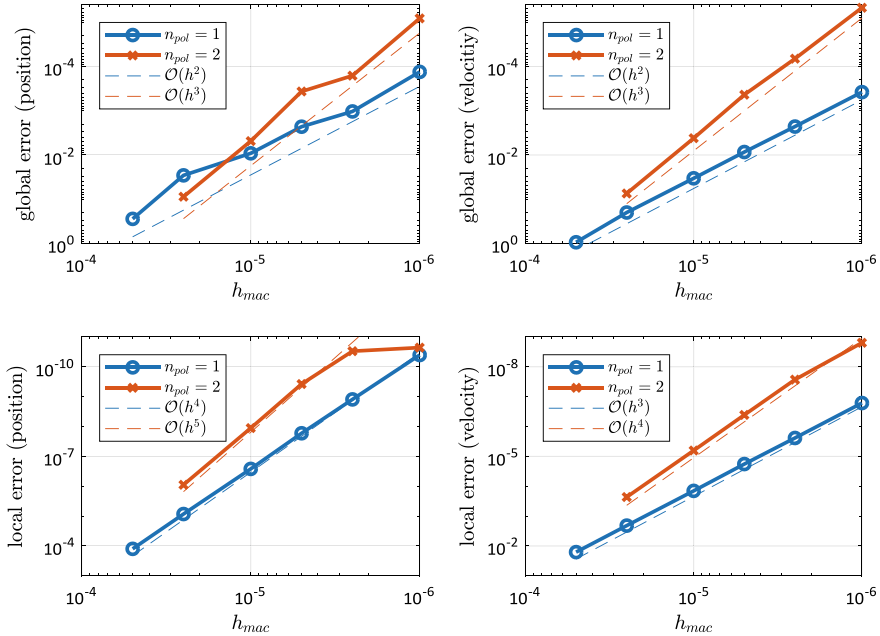


Fig. 8.10 Convergence analysis, explicit co-simulation (subsystem error tolerance $1.0e-12$)

$$F_{Si} = \frac{1}{2} \Delta F_{Si} \left[\tanh\left(\frac{t - t_i}{\delta}\right) - \tanh\left(\frac{t - (t_i + \delta)}{\delta}\right) \right].$$

This may be considered as an approximation of an impact force acting on body i over a short time interval Δt with an amplitude ΔF_{Si} at t_i . The initial positions and velocities of the bodies are chosen randomly in the interval $[-0.1, +0.1]$ and $[-100, +100]$.

8.4.1 Convergence Analysis

Explicit Co-Simulation Method. The convergence behavior of the explicit co-simulation method is investigated by varying the (constant) macro-step size and by evaluating the global and the local error of the state variables.

Figure 8.10 shows the results for the convergence analysis of model 1. Co-simulations are carried out with linear (blue curve) and quadratic (red curve) approximation polynomials for the coupling variables. The subsystem solver tolerance is set to $1.0e-12$ to minimize the numerical errors that are introduced by the subsystem solvers. As can be seen, the local error converges with order $n_{pol} + 3$ on position level and with order $n_{pol} + 2$ on velocity level. The global error converges with $n_{pol} + 1$.

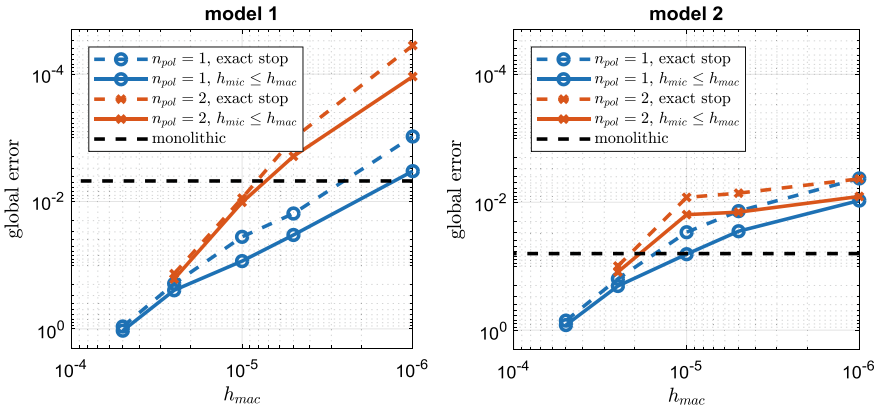


Fig. 8.11 Convergence analysis, explicit co-simulation (subsystem error tolerance $1.0e-6$)

When the subsystem solver tolerance is increased to $1.0e-6$, the overall accuracy of the co-simulation is limited by the subsystem errors. This can be clearly observed for a co-simulation of model 2 with quadratic approximation polynomials for the coupling variables (Fig. 8.11 right hand side, red curve), where the error remains almost constant when the macro-step size is decreased below $1.0e-5$. Due to the micro-step size restriction, the subsystem accuracy of model 1 is implicitly increased for small macro-step sizes; therefore, this effect cannot be observed very clearly for model 1.

The dashed curves in Fig. 8.11 show the error of a co-simulation with the restriction that the subsystem solver stops exactly at each macro-time point. As expected, a co-simulation with this restriction produces slightly lower errors than a co-simulation with the limitation $h_{mic} \leq h_{mac}$. The black dashed lines in Fig. 8.11 show the errors of monolithic simulations of both models for comparison.

Semi-Implicit Co-Simulation Method. The convergence order of the semi-implicit co-simulation method is the same as for the explicit method. The advantages of the semi-implicit approach are the smaller errors and the increased numerical stability.

Figure 8.12 shows a comparison of the numerical errors of the explicit (dashed curves) and the semi-implicit co-simulation method. As can be seen, the error of the semi-implicit method is for both models significantly smaller, especially for large macro-step sizes. In addition, the macro-step size can be increased to $h_{mac} = 7.5e-5$ for the semi-implicit method, while the explicit method needs a macro-step size of $h_{mac} \leq 5.0e-5$ with linear approximation polynomials or $h_{mac} \leq 2.5e-5$ with quadratic polynomials in order to achieve a stable co-simulation. The black dashed lines in Fig. 8.12 show the error of monolithic simulations of the two models for comparison. The monolithic simulations are carried out with an error tolerance of $1.0e-6$, the same error tolerance is used for the subsystem.

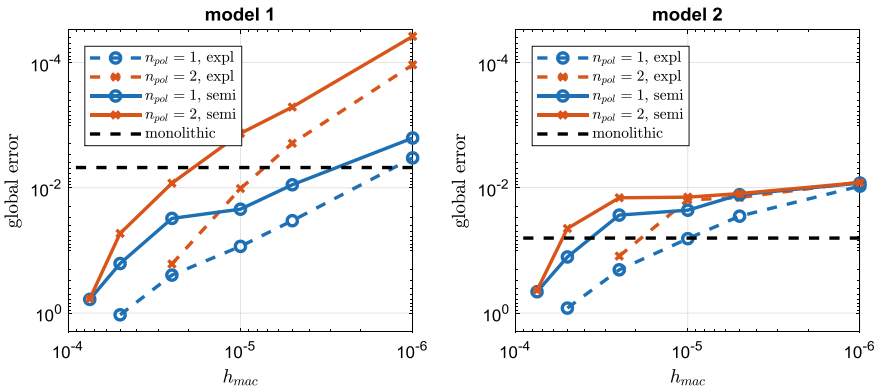


Fig. 8.12 Convergence analysis, semi-implicit co-simulation (subsystem error tolerance $1.0e-6$)

8.4.2 Computation Time Analysis

Computation Time of the Monolithic Simulation. As mentioned before, the scaling factor of the computation time of a multibody system with respect to the number of degrees of freedom is typically between 1 and 3. There exist, for instance, recursive $\mathcal{O}(n)$ algorithms [10] that scale linear. Generally, when the equations of motion are formulated in absolute coordinates and the system is integrated numerically with a BDF method, the dominating factor in the computation time is the linear system that has to be solved within the Newton iterations. The system matrices in our test cases are sparse, because of the simple structure of the test model. This sparsity is exploited by the KLU linear solver resulting in an almost linear scaling of the computation time for model 1. Model 2 is—because of the excitation by the impact forces—stronger affected by the nonlinearities. Therefore, more Newton iterations are needed in each solver step.

Since the scaling factor of the computation time with respect to the degrees of freedom is higher for model 2—at least within the considered range of degrees of freedom—we expect also a greater benefit of a parallel co-simulation for model 2. Figure 8.13 shows the computation time of monolithic simulations of the two test models. The parameters are given in Table 8.1, only the number of masses n_K is varied.

Influence of the Number of Subsystems on the Computation Time. The choice of the number of subsystems, in which a model should be subdivided for achieving the minimal computation time, depends on several aspects. It can be restricted by the topology of the model or limited by the available computer architecture. When there are no restrictions, it is a tradeoff between the reduction of the computation time of each subsystem due to the reduced subsystem size and the cost of the synchronization and data transfer between the subsystems.

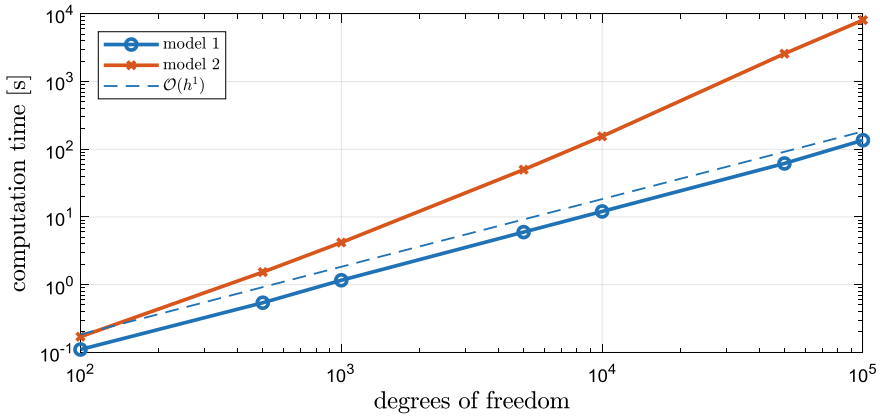


Fig. 8.13 Scaling of the computation time with the number of degrees of freedom

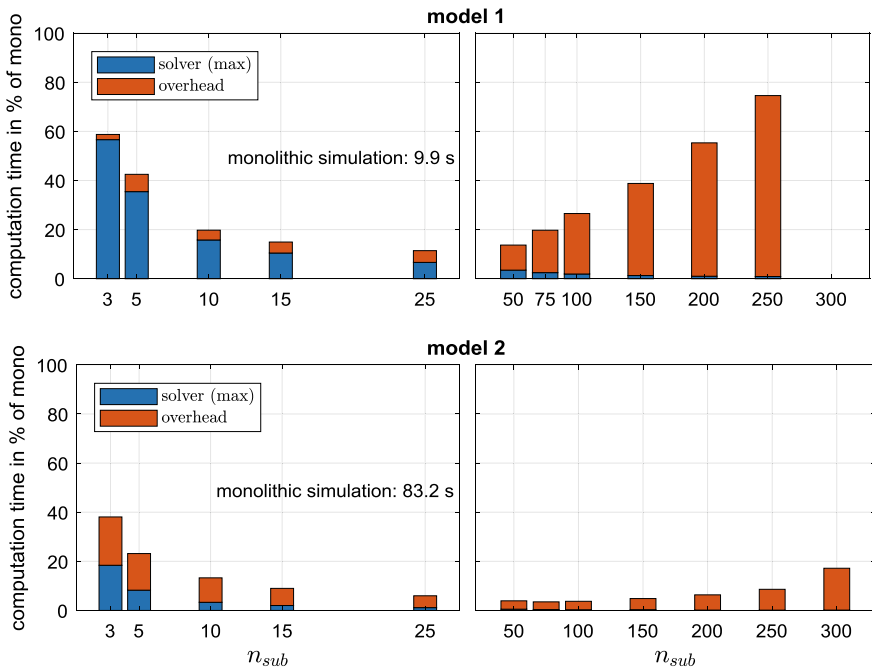


Fig. 8.14 Explicit co-simulation: computation time depending on the number of subsystems

Figure 8.14 shows the computation time of explicit co-simulations ($n_{pol} = 2, h_{mac} = 1.0e - 5, h_{mic} \leq h_{mac}$) with a various number of subsystems. The blue bars show the computation time of the subsystem solver and the red bars the overhead. As expected, the solver time per subsystem decreases as the number of subsystems increases.

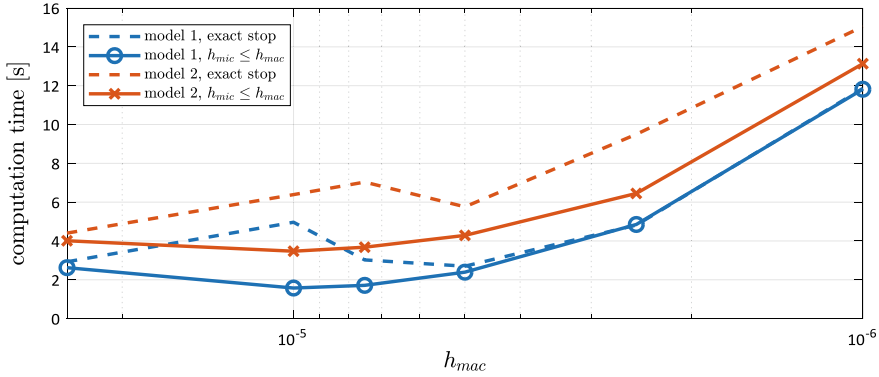


Fig. 8.15 Explicit co-simulation: computation time as a function of the macro-step size

The overhead that appears in co-simulations of model 2 with a small number of subsystems results from different subsystem integration times within each macro-step (cf. Fig. 8.9b). For model 1, the subsystem solver usually makes one micro-step per macro-step (cf. Fig. 8.9c). Therefore, the integration times of all subsystems are almost equal in each macro-step. The number of micro-steps for model 1 is—mainly because of the micro-step size limitation—almost doubled compared to the monolithic simulation. Hence, a co-simulation with three subsystems takes about 60% of the computation time of the monolithic simulation instead of the 33% that would be expected.

When the number of subsystems is increased over a certain level, the overhead caused by network traffic and synchronization becomes the dominating factor of the overall computation time.

Influence of the Macro-Step Size on the Computation Time. The following results are obtained by co-simulations with 47 subsystems. Figure 8.15 shows the computation time as a function of the macro-step size of explicit co-simulations of both test models with quadratic approximation polynomials for the coupling variables. The qualitative observation is the same for both models. When the limitation $h_{mic} \leq h_{mac}$ is applied, the computation time decreases as the macro-step size is decreased until it reaches a minimum. This is the optimal macro-step size, at which the discontinuities in the coupling variables are small and the macro-step size is large enough to not interfere with the micro-step size. When the macro-step size is decreased further, the micro-step size will be restricted by the macro-step size resulting in an increased computation time.

The computation time (Fig. 8.15 dashed curves) of co-simulations with the restriction that the subsystem integration stops at each macro-time point exactly behaves different. In this case, the key point is that the step size controller of the subsystems solver is interrupted at each macro-time point. Therefore, the computation time increases as the macro-step size is decreased. The optimal macro-step size is when

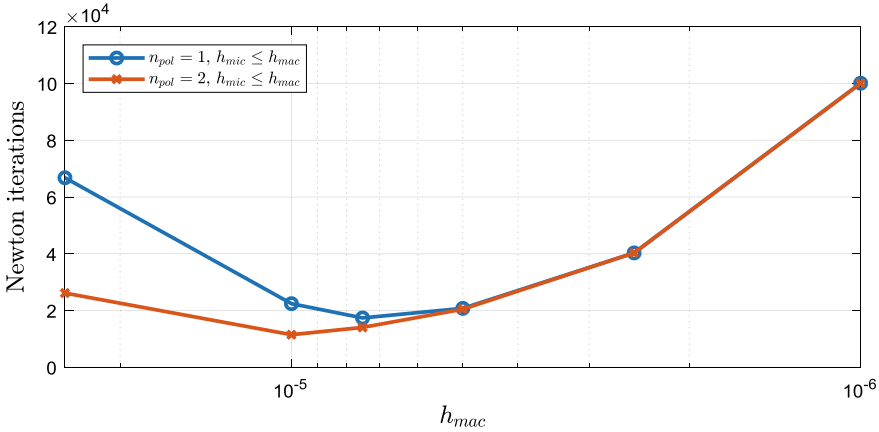


Fig. 8.16 Effect of the discontinuities in the coupling variables on the number of Newton iterations of the subsystem solver

the subsystem solver takes one micro-step per macro-step. When the macro-step size is reduced beyond this point, the computation time increases again.

To clarify the influence of the discontinuities in the coupling variables that result from a large macro-step size, Fig. 8.16 shows the average number of Newton iterations of the subsystem solvers for the co-simulation of model 1. If linear approximation polynomials for the coupling variables are used, the effect of the discontinuities on the computation time is stronger. The average number of Newton iterations for an explicit co-simulation with a macro-step size of $2.5e-5$ is more than 3 times larger than for a co-simulation with a macro-step size of $7.5e-6$.

As mentioned above, the discontinuities in the coupling variables can be reduced or even removed by modifying the approximation polynomials. Figure 8.17 shows the computation time of model 1 (explicit co-simulation, quadratic approximation order) as a function of the macro-step size. Both smoothing approaches, namely the extrapolated interpolation (“extripol”) [9] and the modified interpolation described in Sect. 8.3.3 (“modip”), reduce the computation time for large macro-step sizes significantly.

Macro-Step Size Controller. Figure 8.18 shows the number of macro-steps of an explicit co-simulation with quadratic approximation polynomials. The co-simulation with a constant macro-step size is carried out with $h_{mac} = 1.0e-5$. The error tolerances of the macro-step size controller are chosen in such a way that the global numerical error of both simulations is of the same order of magnitude. The usage of the macro-step size controller reduces the number of macro-steps for both models significantly. For model 2 the number of macro-steps is reduced by about 25% and for model 1 it is reduced by more than 50%. Especially for co-simulations where the dominating factor of the computation time is the data transfer between the subsystems, the macro-step size controller may reduce the computation time significantly.

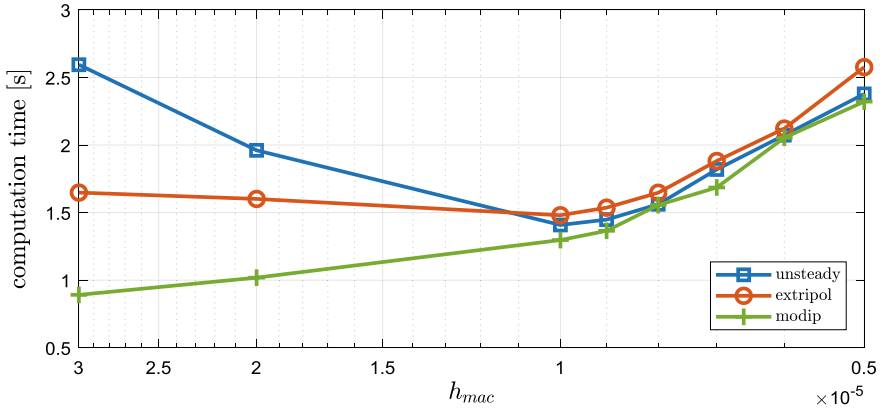


Fig. 8.17 Explicit co-simulation with smoothed interpolation

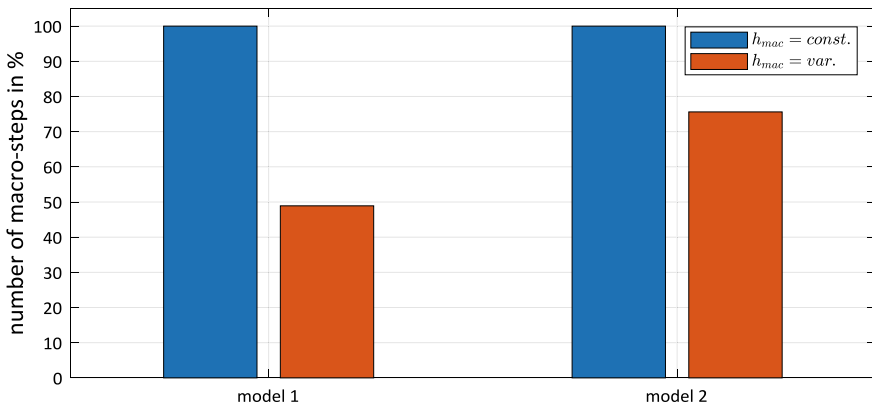


Fig. 8.18 Explicit co-simulation with macro-step size controller: number of macro-steps

8.5 Conclusions

The analysis and the reduction of the computation time using a parallel implementation of a Jacobi-type co-simulation is a challenging task because it is influenced by many factors. Besides the computational aspects like data transfer and thread synchronization in parallel computing, also the effect of the particular co-simulation method on the subsystem solvers plays an important role.

In this manuscript, two co-simulation methods, namely an explicit and a semi-implicit method have been utilized to parallelize the computation of a multibody system. The convergence behavior has been analyzed in detail. Although both methods have the same convergence order, the errors of the semi-implicit co-simulation approach are significantly smaller than for the explicit approach, especially for larger macro-step sizes. The semi-implicit method shows also a better numerical stability.

The main drawback of the semi-implicit co-simulation method is that each macro-step has to be repeated and in addition, partial derivatives with respect to the coupling variables have to be computed. For the explicit co-simulation method, a macro-step size controller based on an error estimator has been implemented. It has been shown that the number of macro-steps can be reduced significantly without increasing the numerical error with the help of a macro-step size controller.

In addition, a detailed computation time analysis of the co-simulation methods has been carried out. The effect of different parameters on the overall computation time has been studied. It was pointed out, that the choice of an appropriate macro-step size is a delicate matter because it affects all parts of the overall computation time. Depending on the particular model, the macro-step size may be chosen rather large to minimize the computational effort for data traffic and thread synchronization. For other models, for example our test model, it may be advantageous to choose a smaller macro-step size in order to reduce the discontinuities in the coupling variables and therefore reduce the subsystem computation time. Also, the macro-step size should be chosen large enough so that it does not restrict the micro-step size. It has been shown, that the computation time can be reduced by using modified approximation polynomials to reduce the effect of discontinuities. Assuming that each subsystem is attached to a fixed number of cores, the hardware utilization is affected by the macro-step size, too. A suitable restriction for the micro-step size is $h_{mic} \leq h_{mac}$. The alternative, namely to stop the subsystem solver at each macro-time point exactly, is not practicable because of its negative effect on the subsystem computation time. Another important point is the choice of the number of subsystems. A proper number of subsystems is always a compromise between reducing the subsystem integration time on the one hand, and increasing overhead due to data traffic and thread synchronization on the other hand.

It has been shown, that the computation time of nonlinear multibody systems can be reduced significantly without increasing the numerical error by applying a parallel co-simulation.

References

1. Ambrosio, J., Pombo, J., Pereira, M., Antunes, P., Mosca, A.: A computational procedure for the dynamic analysis of the catenary-pantograph interaction in high-speed trains. *J. Theor. Appl. Mech.* **50**(3), 681–699 (2012) (Warsaw)
2. Naya, M., Cuadrado, J., Dopico, D., Luginis, U.: An efficient unified method for the combined simulation of multibody and hydraulic dynamics: comparison with simplified and co-integration approaches. *Arch. Mech. Eng.* **LVIII**, 223–243 (2011)
3. Kübler, R., Schiehlen, W.: Two methods of simulator coupling. *Math. Comput. Modell. Dyn. Syst.* **6**, 93–113 (2000)
4. Arnold, M.: Stability of sequential modular time integration methods for coupled multibody system models. *J. Comput. Nonlinear Dyn.* **5**, 1–9 (2010)
5. Schweizer, Bernhard, Li, Pu, Daixing, Lu: Explicit and implicit cosimulation methods: stability and convergence analysis for different solver coupling approaches. *J. Comput. Nonlinear Dyn.* **10**(5), 051007 (2015)

6. Meyer, T., Kraft, J., Li, P., Lu, D., Schweizer, B.: Error estimation approach for controlling the macro step-size for explicit co-simulation methods. In: Proceedings of the 7th GACM Colloquium on Computational Mechanics for Young Scientists from Academia and Industry, 11–13 Oct, Stuttgart, Germany (2017)
7. Hindmarsh, A.C., Brown, P.N., Grant, K.E., Lee, S.L., Serban, R., Shumaker, D.E., Woodward, C.S., Collier, A.: SUNDIALS: suite of nonlinear and differential/algebraic equation solvers. *ACM Trans. Math. Softw.* **31**(3) (2005)
8. Davis, T.A., Palamadai Natarajan, E.: Algorithm 907: KLU, a direct sparse solver for circuit simulation problems. *ACM Trans. Math. Softw.* **37.3**, 36 (2010)
9. Busch, M.: Continuous approximation techniques for co-simulation methods: Analysis of numerical stability and local error. *ZAMM-J. Appl. Math. Mech./Zeitschrift für Angewandte Mathematik und Mechanik* **96**(9), 1061–1081 (2016)
10. Featherstone, Roy: The calculation of robot dynamics using articulated-body inertias. *Int. J. Robot. Res.* **2**(1), 13–30 (1983)

Chapter 9

Explicit Co-simulation Approach with Improved Numerical Stability



Pu Li, Daixing Lu, Robert Schmoll and Bernhard Schweizer

Abstract For coupling different solvers, explicit co-simulation approaches are frequently applied, especially when proprietary software tools have to be coupled without full solver access. Explicit solver coupling approaches are usually much simpler to implement than implicit methods. A major drawback of explicit coupling approaches is their reduced numerical stability behavior. Applying classical explicit co-simulation techniques, the coupling variables are approximated by extrapolation/interpolation polynomials in order to carry out the subsystem integration. Typically, Lagrange polynomials are applied for generating the approximation polynomials, using the coupling variables at the macro-time points as sampling points. In this manuscript, an explicit coupling approach is presented, which shows an improved numerical stability behavior. The key idea of the proposed method is to apply polynomial approximation to predict the acceleration variables of the coupling bodies. By integrating the predicted accelerations, one obtains predicted state variables for the coupling bodies, which are used to predict the coupling variables by making use of the constitutive equations of the coupling element. Compared to classical coupling techniques, where the coupling variables are directly approximated, the approach presented here based on approximated accelerations exhibits a significantly improved numerical stability behavior. Moreover, the numerical error is markedly reduced. The co-simulation method is introduced for flexible multibody systems. However, the proposed approach can generally be applied to couple arbitrary mechanical systems. Also, the coupling technique may be used to couple non-mechanical systems, e.g. electrical or hydraulic systems.

P. Li (✉) · D. Lu · R. Schmoll · B. Schweizer
Technische Universität Darmstadt, Otto-Bernd-Str. 2, 64287 Darmstadt, Germany
e-mail: lipu1215@xjtu.edu.cn

D. Lu
e-mail: lu@ad.tu-darmstadt.de

R. Schmoll
e-mail: schmoll@ad.tu-darmstadt.de

B. Schweizer
e-mail: schweizer@ad.tu-darmstadt.de

9.1 Introduction

Improving the numerical analysis of dynamical systems with respect to accuracy and efficiency is an ongoing topic of research. Different possibilities exist in order to enhance transient simulations. In the field of flexible multibody dynamics, frequently BDF discretization schemes or implicit Runge-Kutta methods are used for integrating the governing system of differential-algebraic equations (DAE system). Using alternative numerical time integration schemes may be one option to improve the efficiency of the numerical simulation. Recently, integration schemes well-established in the field of finite-element analysis—e.g. Newmark methods and related algorithms [17–20, 23, 24, 59]—have been extended and applied for solving the governing DAEs for flexible multibody systems. Furthermore, time integration schemes with special features—like energy and momentum preserving algorithms [11–16, 48–50]—have been developed. To improve the efficiency of the dynamic analyses of large-scaled multibody systems, parallelization may be considered as the most important way to significantly reduce the computation time. Special algorithms have been developed for the parallel implementation of multibody systems. These algorithms pose, however, often special requirements on the structure of the multibody system (e.g. tree-structured system), see for instance Refs. [5, 6, 35].

Co-simulation can be considered as a general approach to couple two (or more) subsystems in time domain. On the one hand, co-simulation techniques are frequently applied to couple two (or several) different software tools in order to simulate multi-physical or multidisciplinary systems. On the other hand, co-simulation approaches can also be used to parallelize (monodisciplinary) dynamical models. Using co-simulation techniques for parallelizing dynamic systems, there are in general no special requirements with respect to the structure of the multibody system [25, 46, 54, 60, 62, 63, 76].

In practical applications, explicit co-simulation methods are frequently used to couple different subsystem solvers. Although implicit coupling methods exhibit a significantly better numerical stability behavior, application of implicit coupling approaches may be problematic, especially if commercial solvers have to be coupled. Implicit methods have the general drawback that the macro-steps have to be repeated. Therefore, the subsystem solvers have to be reinitialized at the previous macro-time point. Depending on the coupling technique, implicit methods often require Jacobian information, which may not easily be obtained if commercial simulation tools are involved. For that reason, explicit methods are often favored and there is a great interest and a need for explicit co-simulation methods with improved stability and convergence properties.

To extrapolate/interpolate the coupling variables, frequently Lagrange polynomials are applied. In this case, the coupling variables at the macro-time points are used to generate extrapolation/interpolation polynomials. In the paper at hand, a different approach is used for approximating the coupling variables. The basic idea is to omit a direct approximating of the coupling variables. Instead of this the acceleration variables of the two coupling bodies are approximated by Lagrange polynomials. An

analytical integration of the approximated accelerations yields approximated velocity and position variables. With the approximated velocity and position variables and with the help of the constitutive equations for the coupling forces/torques—i.e. by means of the physical laws describing the relationship between the coupling forces/torques and the position/velocity variables of the two coupling bodies—approximated coupling variables can be calculated. Using the approximated coupling variables, the subsystems are integrated over the macro-step. Making use of the modified approximation technique, the numerical stability and the convergence behavior of explicit co-simulation approaches may be enhanced considerably. As will be shown in the manuscript, the usage of constant polynomials for approximating the accelerations yields already good results with respect to the numerical stability and with regard to the numerical error.

In order to couple two subsystem solvers, two basic approaches can be applied. Firstly, the subsystem can be coupled by applied forces/torques [1, 4, 28, 37, 44, 52, 61, 69]. In this case, constitutive equations for the coupling forces/torques are used to describe the connection between the subsystems (e.g. spring/damper laws for the coupling forces/torques). Secondly, two subsystems may be coupled by reaction forces/torques [40, 45, 68, 73, 75, 81, 83]. Then, algebraic equations are used to define the coupling between the subsystems (e.g. constraint equations for rigid joints). While explicit methods may be advantageously used in connection with applied-force/torque coupling, for stability reasons usually implicit methods are applied in connection with constraint coupling. Stability and convergence of co-simulation methods based on applied forces/torques have, for instance, been examined in Refs. [34, 37, 51, 57, 71, 74, 80]; coupling techniques on the basis of algebraic constraint equations have been investigated in Refs. [8, 53, 70, 72, 77] with respect to stability and convergence. In this manuscript, only solver coupling based on constitutive equations is considered.

The basic idea behind a co-simulation approach is to decompose an overall dynamical system into different subsystems. Here, only decomposition into two subsystems is considered for the reason of a clear representation. Decomposition into multiple subsystems can be accomplished in a similar manner. It should be mentioned that basically three different decomposition techniques can be distinguished: force/force-, force/displacement- and displacement/displacement-decomposition. The connection between the decomposed subsystems is described by appropriate coupling variables (subsystem input/output variables). In the framework of a co-simulation approach (weak coupling approach) [36], the coupling variables are only interchanged at the macro-time points T_0, T_1, \dots, T_N and the subsystem solvers integrate independently between the macro-time points. In this paper, the macro-step size $H = T_{N+1} - T_N$ is assumed to be constant. Non-equidistant communication-time grids are, for instance, discussed in Refs. [9, 22, 82]. For carrying out the subsystem integration, the coupling variables have to be approximated between the macro-time points. Using a parallel subsystem integration (Jacobi type), both subsystems are integrated with extrapolated coupling variables. Applying a serial integration scheme (Gauss-Seidel type), the first subsystem is integrated with extrapolated coupling variables, while interpolated coupling variables are used for the integration of the second subsystem.

The efficient and robust simulation of complex multiphysical systems makes high demands on the software tools and the numerical methods. Instead of simulating complex systems with only one simulation tool and a single solver, it is often more convenient to couple different specialized software tools in order to carry out multiphysical and multidisciplinary simulations. Therefore, accurate and efficient solver coupling techniques have to be used. Solver coupling is frequently applied to analyze fluid/structure interaction problems, see for instance Refs. [1, 29, 55, 57, 65, 67]. Co-simulation techniques are also often used to couple multibody models with finite-element or boundary-element models, see Refs. [2–4, 21, 85, 86]. The coupled simulation of particle models with multibody models is treated in Refs. [31, 47, 79], for instance. Co-simulation in context with non-smooth dynamics is discussed in Refs. [32, 33]. Solver coupling of multibody systems with block diagram simulators is analyzed in Ref. [38]. Solver coupling in connection with electromechanical problems has, for instance, been investigated in Refs. [43, 84]. Application of co-simulation methods in the field of vehicle dynamics is treated in [27, 28, 30, 52, 61]. Besides the coupled simulation of multiphysical systems, co-simulation techniques may also be applied for parallelizing monodisciplinary dynamical models, as mentioned above. It should be pointed out that explicit co-simulation approaches show a certain similarity to subcycling methods, which are frequently used to speed up the simulation of finite-element models by splitting the system into different parts, which are integrated with different time-step sizes [10, 26, 56, 58]. In the framework of a co-simulation approach, the subsystems are usually regarded as black box systems. It is desired that a co-simulation method can be applied to arbitrary subsystems, independent from the subsystem solvers. Classical subcycling methods are, however, typically used in connection with explicit integration schemes (central difference method) and the subcycling algorithm is embedded into the integration scheme of the subsystems.

The main contributions of this manuscript are:

- Developing an alternative explicit co-simulation approach. Considering the well-established co-simulation methods known from literature, where position and velocity variables are approximated usually by Lagrange polynomials, the coupling technique applied in the paper at hand is based on approximated acceleration variables. One advantage of the presented approach is that the method may be implemented as a self-starting algorithm yielding quadratic convergence order. Compared to the established approaches, the presented method has the drawback that an update step for the acceleration variables is required after each macro-time step.
- Providing a stability analysis which shows the improved stability behavior of the proposed approach with respect to classical explicit co-simulation methods.
- Comparing the convergence behavior of the new coupling approach with classical explicit co-simulation methods.
- Demonstrating the practical applicability of the proposed approach with nonlinear test models and large-scaled industrial models.

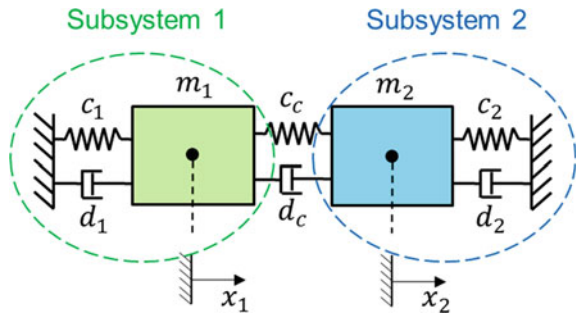
This manuscript is structured as follows: An explicit co-simulation method based on approximated accelerations is introduced in Sect. 9.2 with the help of a co-simulation test model. The numerical stability and the convergence behavior of the integrated acceleration co-simulation approach is in detail investigated for the three possible decomposition techniques (force/force-, force/displacement- and displacement/displacement-decomposition) considering both parallel (Jacobi type) and sequential (Gauss-Seidel type) integration. The integrated acceleration approach is compared with the classical explicit co-simulation approach based on extrapolated coupling variables. The generalization of the integrated acceleration co-simulation approach for coupling two arbitrary mechanical systems is treated in Sect. 9.3. Numerical examples are collected in Sect. 9.4. The results are summarized in Sect. 9.5. Detailed stability and convergence plots for the case that the accelerations are approximated by linear polynomials are collected in Appendix 1. Some additional considerations on the sequential force/displacement-decomposition approach can be found in Appendix 2.

9.2 An Explicit Co-simulation Method

For analyzing the numerical stability of time integration schemes (Runge-Kutta methods, BDF methods, etc.), Dahlquist’s test equation is used [41, 42], which can—from the mechanical point of view—be interpreted as the complex representation of the homogenous linear single-mass oscillator. To analyze the stability of co-simulation methods, an appropriate test model has to be defined. The most appropriate test model for co-simulation methods based on applied-force/torque coupling is the homogenous linear two-mass oscillator, see Fig. 9.1. The two-mass oscillator can be interpreted as two single mass-oscillators (masses m_1/m_2 , springs c_1/c_2 and dampers d_1/d_2), which are coupled by the coupling spring c_c and the coupling damper d_c . From the mathematical point of view, the co-simulation test model can be considered as two Dahlquist equations, coupled by a linear constitutive equation.

x_1, x_2 and v_1, v_2 denote position and velocity of the two masses. For the subsequent stability analysis, it is useful to introduce dimensionless variables. We assume that

Fig. 9.1 Homogenous linear two-mass oscillator: test model for analyzing the stability of co-simulation methods



\bar{x}_1, \bar{x}_2 are properly chosen dimensionless position coordinates. The variables $\bar{v}_1 = H \cdot \frac{d\bar{x}_1}{dt}$ and $\bar{v}_2 = H \cdot \frac{d\bar{x}_2}{dt}$ denote dimensionless velocities, where H characterizes the communication-step size of the coupling approach. The dimensionless time is defined by $\bar{t} = \frac{t}{H}$. Furthermore, the following 7 parameters are introduced

$$\begin{aligned} \bar{c}_1 &= \frac{c_1 \cdot H^2}{m_1}, \bar{d}_1 = \frac{d_1 \cdot H}{m_1}, \alpha_{m21} = \frac{m_2}{m_1}, \alpha_{c21} = \frac{c_2}{c_1}, \alpha_{d21} = \frac{d_2}{d_1}, \\ \alpha_{cc1} &= \frac{c_c}{c_1}, \alpha_{dc1} = \frac{d_c}{d_1}. \end{aligned} \quad (9.1)$$

Then, the equations of motion for the two-mass oscillator can in dimensionless form be written by

$$\begin{aligned} \bar{x}'_1 &= \bar{v}_1 \\ \bar{v}'_1 &= -\bar{c}_1 \cdot \bar{x}_1 - \bar{d}_1 \cdot \bar{v}_1 + \alpha_{cc1} \cdot \bar{c}_1 \cdot (\bar{x}_2 - \bar{x}_1) + \alpha_{dc1} \cdot \bar{d}_1 \cdot (\bar{v}_2 - \bar{v}_1) \\ \bar{x}'_2 &= \bar{v}_2 \\ \bar{v}'_2 &= -\frac{\alpha_{c21}}{\alpha_{m21}} \cdot \bar{c}_1 \cdot \bar{x}_2 - \frac{\alpha_{d21}}{\alpha_{m21}} \cdot \bar{d}_1 \cdot \bar{v}_2 - \frac{\alpha_{cc1}}{\alpha_{m21}} \cdot \bar{c}_1 \cdot (\bar{x}_2 - \bar{x}_1) \\ &\quad - \frac{\alpha_{dc1}}{\alpha_{m21}} \cdot \bar{d}_1 \cdot (\bar{v}_2 - \bar{v}_1). \end{aligned} \quad (9.2)$$

Note that $(\cdot)' = \frac{d(\cdot)}{d\bar{t}}$ represents the derivative with respect to the dimensionless time \bar{t} .

Obviously, the two-mass oscillator is a mechanically stable system, if $m_1, m_2, c_1, c_2, c_c, d_1, d_2, d_c > 0$. Decomposing the two-mass oscillator into two subsystems and discretizing Eq. (9.2) by a linear co-simulation approach, a homogeneous linear system of recurrence equations is obtained, see Sects. 9.2.1–9.2.3. The stability of the recurrence system defines the numerical stability of the underlying co-simulation approach. If the spectral radius ρ of the recurrence system is smaller than 1, the co-simulation approach is called numerically stable. The spectral radius ρ is a nonlinear function of the 7 independent test model parameters defined in Eq. (9.1). For a graphical illustration of the numerical stability of co-simulation methods, 2D stability plots are most appropriate. Therefore, it is useful to fix 5 of the entire 7 test model parameters and to plot the spectral radius ρ as a function of the remaining two parameters.

The decomposition of the two-mass oscillator into two subsystems can be accomplished by three different techniques, namely by a force/force-, a force/displacement- or a displacement/displacement-decomposition approach. Concerning the subsystem integration, two basic integration types may be distinguished, namely parallel (Jacobi type) and sequential (Gauss-Seidel type) integration. In the next subsections, the integrated acceleration co-simulation approach is derived and analyzed with respect to numerical stability and convergence behavior for all three decomposition types and for both integration types.

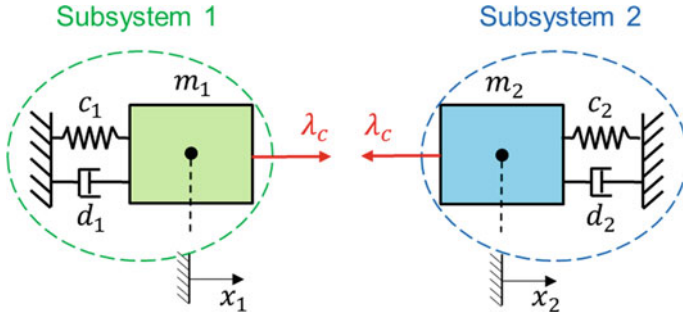


Fig. 9.2 Co-simulation test model: force/force-decomposition approach

9.2.1 Force/Force-Coupling

Parallel Integration (Jacobi Type). Making use of a force/force-decomposition approach, the co-simulation test model is split into two subsystems so that both subsystems are force-driven single-mass oscillators, see Fig. 9.2. The coupling force λ_c is given by $\lambda_c = c_c \cdot (x_2 - x_1) + d_c \cdot (v_2 - v_1)$.

With the dimensionless position and velocity variables, with the dimensionless coupling force $\bar{\lambda}_c = \frac{\lambda_c H^2}{m_1}$ and with the parameters of Eq. (9.1), the decomposed system is described by the following system of equations

$$\begin{aligned}
 &\text{Subsystem 1:} \\
 &\quad \bar{x}'_1 = \bar{v}_1 \\
 &\quad \bar{v}'_1 = -\bar{c}_1 \cdot \bar{x}_1 - \bar{d}_1 \cdot \bar{v}_1 + \bar{\lambda}_c \tag{a} \\
 &\text{Subsystem 2:} \\
 &\quad \bar{x}'_2 = \bar{v}_2 \\
 &\quad \bar{v}'_2 = -\frac{\alpha_{c21}}{\alpha_{m21}} \cdot \bar{c}_1 \cdot \bar{x}_2 - \frac{\alpha_{d21}}{\alpha_{m21}} \cdot \bar{d}_1 \cdot \bar{v}_2 - \frac{1}{\alpha_{m21}} \bar{\lambda}_c \tag{b} \\
 &\text{Coupling condition:} \\
 &\quad \bar{\lambda}_c - \alpha_{cc1} \cdot \bar{c}_1 \cdot (\bar{x}_2 - \bar{x}_1) - \alpha_{dc1} \cdot \bar{d}_1 \cdot (\bar{v}_2 - \bar{v}_1) = 0. \tag{c}
 \end{aligned}$$

To explain the co-simulation approach, we consider the general macro-time step from \bar{T}_N to \bar{T}_{N+1} ($\bar{T}_N = \frac{T_N}{H}$ denotes the dimensionless macro-time point). At the beginning of the macro-time step, the state variables, the accelerations and the coupling variable are assumed to be known

$$\begin{aligned}
 \bar{x}_1(\bar{t} = \bar{T}_N) &= \bar{x}_{1,N}, \quad \bar{v}_1(\bar{t} = \bar{T}_N) = \bar{v}_{1,N}, \quad \bar{v}'_1(\bar{t} = \bar{T}_N) = \bar{v}'_{1,N}, \tag{a} \\
 \bar{x}_2(\bar{t} = \bar{T}_N) &= \bar{x}_{2,N}, \quad \bar{v}_2(\bar{t} = \bar{T}_N) = \bar{v}_{2,N}, \quad \bar{v}'_2(\bar{t} = \bar{T}_N) = \bar{v}'_{2,N}, \tag{b} \\
 \bar{\lambda}_c(\bar{t} = \bar{T}_N) &= \lambda_{c,N}. \tag{c}
 \end{aligned}$$

Step 1: Integration of Extrapolated Accelerations

- For the subsystem integration from \bar{T}_N to \bar{T}_{N+1} , the coupling variable $\bar{\lambda}_c(\bar{t})$ has to be approximated in the time interval $[\bar{T}_N, \bar{T}_{N+1}]$. With the acceleration $\bar{v}'_{1,N}$ at the current macro-time point \bar{T}_N and the accelerations $\bar{v}'_{1,N-1}$, $\bar{v}'_{1,N-2}$, etc. at the previous macro-time points \bar{T}_{N-1} , \bar{T}_{N-2} , etc., an extrapolation polynomial (Lagrange polynomial) $\bar{v}'_1{}^{ex}(\bar{t})$ of degree k can be generated based on the sampling points $(\bar{T}_N, \bar{v}'_{1,N})$, $(\bar{T}_{N-1}, \bar{v}'_{1,N-1})$, \dots , $(\bar{T}_{N-k}, \bar{v}'_{1,N-k})$. Accordingly, an extrapolation polynomial $\bar{v}'_2{}^{ex}(\bar{t})$ of degree k can be constructed for the acceleration of mass 2 using the sampling points $(\bar{T}_N, \bar{v}'_{2,N})$, $(\bar{T}_{N-1}, \bar{v}'_{2,N-1})$, \dots , $(\bar{T}_{N-k}, \bar{v}'_{2,N-k})$.
- An analytical forward integration of the polynomials $\bar{v}'_1{}^{ex}(\bar{t})$ and $\bar{v}'_2{}^{ex}(\bar{t})$ from \bar{T}_N to \bar{t} ($\bar{t} \in [\bar{T}_N, \bar{T}_{N+1}]$) yields the extrapolation polynomials

$$\bar{v}_1{}^{ex}(\bar{t}) = \int_{\bar{T}_N}^{\bar{t}} \bar{v}'_1{}^{ex}(s) ds, \quad \bar{v}_2{}^{ex}(\bar{t}) = \int_{\bar{T}_N}^{\bar{t}} \bar{v}'_2{}^{ex}(s) ds \quad (9.5)$$

of degree $k + 1$ for the velocity variables and the approximation polynomials

$$\bar{x}_1{}^{ex}(\bar{t}) = \int_{\bar{T}_N}^{\bar{t}} \bar{v}_1{}^{ex}(s) ds, \quad \bar{x}_2{}^{ex}(\bar{t}) = \int_{\bar{T}_N}^{\bar{t}} \bar{v}_2{}^{ex}(s) ds \quad (9.6)$$

of degree $k + 2$ for the position variables in the time interval $[\bar{T}_N, \bar{T}_{N+1}]$. As initial conditions for the forward integration, the state variables at the macro-time point \bar{T}_N according to Eq. (9.4) are used, i.e. $\bar{v}_1{}^{ex}(\bar{t} = \bar{T}_N) = \bar{v}_{1,N}$ and $\bar{x}_1{}^{ex}(\bar{t} = \bar{T}_N) = \bar{x}_{1,N}$ as well as $\bar{v}_2{}^{ex}(\bar{t} = \bar{T}_N) = \bar{v}_{2,N}$ and $\bar{x}_2{}^{ex}(\bar{t} = \bar{T}_N) = \bar{x}_{2,N}$.

- Note that for the special case of constant approximation ($k = 0$), i.e. assuming $\bar{v}'_1{}^{ex}(\bar{t}) = \bar{v}'_{1,N} = \text{const.}$ and $\bar{v}'_2{}^{ex}(\bar{t}) = \bar{v}'_{2,N} = \text{const.}$, the following extrapolated state variables are obtained

$$\begin{aligned} \bar{v}_1{}^{ex}(\bar{t}) &= \bar{v}'_{1,N} \cdot (\bar{t} - \bar{T}_N) + \bar{v}_{1,N}, \\ \bar{x}_1{}^{ex}(\bar{t}) &= \frac{1}{2} \bar{v}'_{1,N} \cdot (\bar{t} - \bar{T}_N)^2 + \bar{v}_{1,N} \cdot (\bar{t} - \bar{T}_N) + \bar{x}_{1,N}, \\ \bar{v}_2{}^{ex}(\bar{t}) &= \bar{v}'_{2,N} \cdot (\bar{t} - \bar{T}_N) + \bar{v}_{2,N}, \\ \bar{x}_2{}^{ex}(\bar{t}) &= \frac{1}{2} \bar{v}'_{2,N} \cdot (\bar{t} - \bar{T}_N)^2 + \bar{v}_{2,N} \cdot (\bar{t} - \bar{T}_N) + \bar{x}_{2,N}. \end{aligned} \quad (9.7)$$

Step 2: Calculation of Extrapolated Coupling Variable

- With the extrapolated state variables, an extrapolation polynomial $\bar{\lambda}_c{}^{ex}(\bar{t})$ of degree $k + 2$ for the coupling force in the time interval $[\bar{T}_N, \bar{T}_{N+1}]$ can be calculated by

$$\bar{\lambda}_c{}^{ex}(\bar{t}) = \alpha_{cc1} \cdot \bar{c}_1 \cdot (\bar{x}_2{}^{ex}(\bar{t}) - \bar{x}_1{}^{ex}(\bar{t})) + \alpha_{dc1} \cdot \bar{d}_1 \cdot (\bar{v}_2{}^{ex}(\bar{t}) - \bar{v}_1{}^{ex}(\bar{t})). \quad (9.8)$$

Step 3: Integration of Subsystems

- An analytical integration of subsystem 1 and subsystem 2 from \bar{T}_N to \bar{T}_{N+1} with initial conditions (9.4) and with the extrapolated coupling force $\bar{\lambda}_c^{ex}(\bar{t})$ according to Eq. (9.8) yields the new state variables at the macro-time point \bar{T}_{N+1}

$$\begin{aligned}\bar{x}_{1,N+1} &= \bar{x}_{1,N+1}(\bar{z}_N, \dots, \bar{z}_{N-k}), & \bar{v}_{1,N+1} &= \bar{v}_{1,N+1}(\bar{z}_N, \dots, \bar{z}_{N-k}), \\ \bar{x}_{2,N+1} &= \bar{x}_{2,N+1}(\bar{z}_N, \dots, \bar{z}_{N-k}), & \bar{v}_{2,N+1} &= \bar{v}_{2,N+1}(\bar{z}_N, \dots, \bar{z}_{N-k}).\end{aligned}\quad (9.9)$$

Note that the auxiliary vectors $\bar{z}_N = (\bar{x}_{1,N}, \bar{v}_{1,N}, \bar{x}_{2,N}, \bar{v}_{2,N})^T$, $\bar{z}_{N-1} = (\bar{x}_{1,N-1}, \bar{v}_{1,N-1}, \bar{x}_{2,N-1}, \bar{v}_{2,N-1})^T$, etc. collect the state variables at the macro-time points.

- Update the accelerations of subsystem 1 and 2 by means of Eq. (9.3)a, b

$$\begin{aligned}\bar{v}'_{1,N+1} &= -\bar{c}_1 \cdot \bar{x}_{1,N+1} - \bar{d}_1 \cdot \bar{v}_{1,N+1} + \bar{\lambda}_{c,N+1}(\bar{z}_{N+1}), \\ \bar{v}'_{2,N+1} &= -\frac{\alpha_{e21}}{\alpha_{m21}} \cdot \bar{c}_1 \cdot \bar{x}_{2,N+1} - \frac{\alpha_{d21}}{\alpha_{m21}} \cdot \bar{d}_1 \cdot \bar{v}_{2,N+1} - \frac{1}{\alpha_{m21}} \bar{\lambda}_{c,N+1}(\bar{z}_{N+1}).\end{aligned}\quad (9.10)$$

Equation (9.9) represents a system of 4 coupled homogenous linear recurrence equations of order $k + 1$, which can symbolically be written as

$$\bar{z}_{N+1} + \mathbf{A}_N \cdot \bar{z}_N + \dots + \mathbf{A}_{N-k} \cdot \bar{z}_{N-k} = \mathbf{0}. \quad (9.11)$$

The real-valued matrices $\mathbf{A}_N, \dots, \mathbf{A}_{N-k} \in \mathbb{R}^{4 \times 4}$ are constant; they depend only on the 7 parameters of the co-simulation test model. Due to the fact that the subsystems are integrated analytically, which is possible due to the fact that the co-simulation test model is linear, the stability of the recurrence system (9.11) directly reflects the numerical stability of the underlying co-simulation approach. The stability of Eq. (9.11) is determined by the spectral radius ρ of the recurrence system. For $\rho \leq 1$, the co-simulation is called numerically stable. Note that ρ is only a function of the 7 test model parameters.

Sequential Integration (Gauss-Seidel Type).

Step 1: Integration of Extrapolated Accelerations

- Identical with step 1 of Jacobi type.

Step 2: Calculation of Extrapolated Coupling Variable

- Identical with step 2 of Jacobi type.

Step 3: Integration of Subsystem 1

- Using a sequential co-simulation approach, subsystem 1 is firstly integrated analytically from \bar{T}_N to \bar{T}_{N+1} with the extrapolated coupling force $\bar{\lambda}_c^{ex}(\bar{t})$ of Eq. (9.8), which gives the new state variables for subsystem 1 at the macro-time point \bar{T}_{N+1}

$$\bar{x}_{1,N+1} = \bar{x}_{1,N+1}(\bar{z}_N, \dots, \bar{z}_{N-k}), \quad \bar{v}_{1,N+1} = \bar{v}_{1,N+1}(\bar{z}_N, \dots, \bar{z}_{N-k}). \quad (9.12)$$

- Update accelerations of subsystem 1 with the help of Eq. (9.3), that is

$$\begin{aligned} \bar{v}'_{1,N+1} = & -\bar{c}_1 \cdot \bar{x}_{1,N+1} - \bar{d}_1 \cdot \bar{v}_{1,N+1} \\ & + \bar{\lambda}_{c,N+1}(\bar{x}_{1,N+1}, \bar{v}_{1,N+1}, \bar{x}_2^{ex}(\bar{T}_{N+1}), \bar{v}_2^{ex}(\bar{T}_{N+1})). \end{aligned} \quad (9.13)$$

Step 4: Integration of Interpolated Accelerations

- Making use of the sampling points $(\bar{T}_{N+1}, \bar{v}'_{1,N+1}), (\bar{T}_N, \bar{v}'_{1,N}), \dots, (\bar{T}_{N-k+1}, \bar{v}'_{1,N-k+1})$, an interpolation polynomial $\bar{v}_1^{in}(\bar{t})$ of degree k for the acceleration of mass 1 can be generated for the time interval $[\bar{T}_N, \bar{T}_{N+1}]$. An analytical backward integration of $\bar{v}_1^{in}(\bar{t})$ from \bar{T}_{N+1} to \bar{t} ($\bar{t} \in [\bar{T}_N, \bar{T}_{N+1}]$) yields the interpolation polynomial

$$\bar{v}_1^{in}(\bar{t}) = \int_{\bar{T}_{N+1}}^{\bar{t}} \bar{v}_1^{in}(s) ds \quad (9.14)$$

of degree $k + 1$ on velocity level and the interpolation polynomial

$$\bar{x}_1^{in}(\bar{t}) = \int_{\bar{T}_{N+1}}^{\bar{t}} \bar{v}_1^{in}(s) ds \quad (9.15)$$

of degree $k + 2$ on position level. As initial conditions for the backward integration, the state variables of Eq. (9.12) have to be used, i.e. $\bar{v}_1^{in}(\bar{t} = \bar{T}_{N+1}) = \bar{v}_{1,N+1}$ and $\bar{x}_1^{in}(\bar{t} = \bar{T}_{N+1}) = \bar{x}_{1,N+1}$.

- For the simple case of constant approximation, the subsequent interpolated state variables are obtained

$$\begin{aligned} \bar{v}_1^{in}(\bar{t}) &= \bar{v}'_{1,N+1} \cdot (\bar{t} - \bar{T}_{N+1}) + \bar{v}_{1,N+1}, \\ \bar{x}_1^{in}(\bar{t}) &= \frac{1}{2} \bar{v}'_{1,N+1} \cdot (\bar{t} - \bar{T}_{N+1})^2 + \bar{v}_{1,N+1} \cdot (\bar{t} - \bar{T}_{N+1}) + \bar{x}_{1,N+1}. \end{aligned} \quad (9.16)$$

Step 5: Calculation of Approximated Coupling Variable

- With the interpolated state variables $\bar{v}_1^{in}(\bar{t}), \bar{x}_1^{in}(\bar{t})$ of Eqs. (9.14) and (9.15), and with the extrapolated state variables $\bar{v}_2^{ex}(\bar{t}), \bar{x}_2^{ex}(\bar{t})$ of Eqs. (9.5) and (9.6), the approximation polynomial

$$\bar{\lambda}_c^{in}(\bar{t}) = \alpha_{cc1} \cdot \bar{c}_1 \cdot (\bar{x}_2^{ex}(\bar{t}) - \bar{x}_1^{in}(\bar{t})) - \alpha_{dc1} \cdot \bar{d}_1 \cdot (\bar{v}_2^{ex}(\bar{t}) - \bar{v}_1^{in}(\bar{t})) \quad (9.17)$$

of degree $k + 2$ for the coupling force can be constructed for the time interval $[\bar{T}_N, \bar{T}_{N+1}]$.

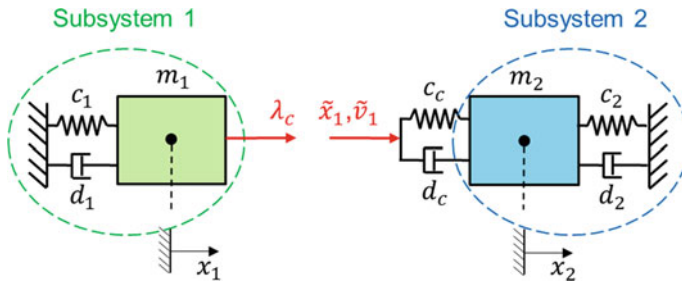


Fig. 9.3 Co-simulation test model: force/displacement-decomposition approach

Step 6: Integration of Subsystem 2

- Finally, subsystem 2 is integrated analytically from \bar{T}_N to \bar{T}_{N+1} with the coupling force $\bar{\lambda}_c^m(\bar{t})$ of Eq. (9.17), which yields

$$\bar{x}_{2,N+1} = \bar{x}_{2,N+1}(\bar{z}_{N+1}, \dots, \bar{z}_{N-k}), \quad \bar{v}_{2,N+1} = \bar{v}_{2,N+1}(\bar{z}_{N+1}, \dots, \bar{z}_{N-k}). \quad (9.18)$$

- Update of the acceleration variables of subsystem 1 and 2 with the help of Eq. (9.3)a, b gives

$$\begin{aligned} \bar{v}'_{1,N+1} &= -\bar{c}_1 \cdot \bar{x}_{1,N+1} - \bar{d}_1 \cdot \bar{v}_{1,N+1} + \bar{\lambda}_{c,N+1}(\bar{z}_{N+1}) \\ \bar{v}'_{2,N+1} &= -\frac{\alpha_{c21}}{\alpha_{m21}} \cdot \bar{c}_1 \cdot \bar{x}_{2,N+1} - \frac{\alpha_{d21}}{\alpha_{m21}} \cdot \bar{d} \cdot \bar{v}_{2,N+1} - \frac{1}{\alpha_{m21}} \bar{\lambda}_{c,N+1}(\bar{z}_{N+1}). \end{aligned} \quad (9.19)$$

Equation (9.12) together with Eq. (9.18) represent a system of 4 coupled homogeneous linear recurrence equations of order $k + 1$ of the form

$$\mathbf{A}_{N+1} \cdot \bar{z}_{N+1} + \mathbf{A}_N \cdot \bar{z}_N + \dots + \mathbf{A}_{N-k} \cdot \bar{z}_{N-k} = \mathbf{0}. \quad (9.20)$$

9.2.2 Force/Displacement-Coupling

Parallel Integration (Jacobi Type). If a force/displacement-decomposition approach is applied, the two-mass oscillator is split into two subsystems so that subsystem 1 will be a force-driven single-mass oscillator and subsystem 2 a base-point excited single-mass oscillator, see Fig. 9.3. In the framework of a force/displacement-approach, the coupling force λ_c has to be replaced in subsystem 2 with the help of the constitutive equation $\lambda_c = c_c \cdot (x_2 - x_1) + d_c \cdot (v_2 - v_1)$. Since the state variables x_1 and v_1 are not available in subsystem 2, they are substituted by the additional coupling variables \bar{x}_1 and \bar{v}_1 . As a consequence, the two additional coupling conditions $\bar{x}_1 - x_1 = 0$ and $\bar{v}_1 - v_1 = 0$ have to be introduced.

In dimensionless form, the equations of motion for the decomposed system read

Subsystem 1:

$$\begin{aligned}\bar{x}'_1 &= \bar{v}_1 \\ \bar{v}'_1 &= -\bar{c}_1 \cdot \bar{x}_1 - \bar{d}_1 \cdot \bar{v}_1 + \bar{\lambda}_c\end{aligned}\quad (a)$$

Subsystem 2:

$$\begin{aligned}\bar{x}'_2 &= \bar{v}_2 \\ \bar{v}'_2 &= -\frac{\alpha_{c21}}{\alpha_{m21}} \cdot \bar{c}_1 \cdot \bar{x}_2 - \frac{\alpha_{d21}}{\alpha_{m21}} \cdot \bar{d}_1 \cdot \bar{v}_2 \\ &\quad - \frac{\alpha_{cc1}}{\alpha_{m21}} \cdot \bar{c}_1 \cdot (\bar{x}_2 - \bar{x}_1) - \frac{\alpha_{dc1}}{\alpha_{m21}} \cdot \bar{d}_1 \cdot (\bar{v}_2 - \bar{v}_1)\end{aligned}\quad (b) \quad (9.21)$$

Coupling conditions:

$$\begin{aligned}\bar{\lambda}_c - \alpha_{cc1} \cdot \bar{c}_1 \cdot (\bar{x}_2 - \bar{x}_1) - \alpha_{dc1} \cdot \bar{d}_1 \cdot (\bar{v}_2 - \bar{v}_1) &= 0 \\ \bar{\tilde{x}}_1 - \bar{x}_1 &= 0 \\ \bar{\tilde{v}}_1 - \bar{v}_1 &= 0.\end{aligned}\quad (c)$$

Again, the general macro-time step from \bar{T}_N to \bar{T}_{N+1} is considered in order to explain the coupling approach. Both subsystems are integrated in parallel. Subsystem 1 is integrated with the extrapolated coupling force $\bar{\lambda}_c = \bar{\lambda}_c^{ex}(\bar{t})$ according to Eq. (9.18). Subsystem 2 is integrated with the extrapolation polynomials $\bar{\tilde{x}}_1(\bar{t}) = \bar{x}_1^{ex}(\bar{t})$ and $\bar{\tilde{v}}_1(\bar{t}) = \bar{v}_1^{ex}(\bar{t})$ according to Eqs. (9.5) and (9.6). Integration of the two subsystems yields the state variables at the new macro-time point \bar{T}_{N+1} similar to Eq. (9.9).

Sequential Integration (Gauss-Seidel Type). As in Sect. 9.2.2 (Jacobi type), Subsystem 1 is integrated with the extrapolated coupling force $\bar{\lambda}_c = \bar{\lambda}_c^{ex}(\bar{t})$ according to Eq. (9.8). Then, Subsystem 2 is integrated using the interpolation polynomials $\bar{\tilde{x}}_1(\bar{t}) = \bar{x}_1^{in}(\bar{t})$ and $\bar{\tilde{v}}_1(\bar{t}) = \bar{v}_1^{in}(\bar{t})$ according to Eqs. (9.14) and (9.15). As a result, we obtain state variables at the new macro-time point \bar{T}_{N+1} analogous to Eqs. (9.12) and (9.18).

It should be mentioned that the sequential force/displacement co-simulation approach bears—for the case that constant approximation polynomials are used for approximating the accelerations—some similarity to the constant acceleration subcycling algorithm introduced in Ref. [10].

9.2.3 Displacement/Displacement-Coupling

Parallel Integration (Jacobi Type). In the framework of a displacement/displacement-coupling approach, the coupling spring/damper element is duplicated so that both subsystems are represented by base-point excited single-mass oscillators, see Fig. 9.4. This is mathematically realized by substituting the coupling force λ_c in both subsystems with the help of the constitutive equation $\lambda_c = c_c \cdot (x_2 - x_1) + d_c \cdot (v_2 - v_1)$. Since the state variables of subsystem 2 are not

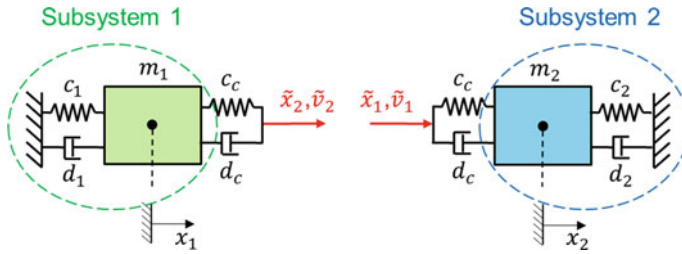


Fig. 9.4 Co-simulation test model: displacement/displacement-decomposition approach

accessible in subsystem 1 (and vice versa), four additional coupling variables and corresponding coupling conditions have to be introduced.

The decomposed system is characterized by the subsequent set of equations

Subsystem 1:

$$\begin{aligned} \bar{x}'_1 &= \bar{v}_1 \\ \bar{v}'_1 &= -\bar{c}_1 \cdot \bar{x}_1 - \bar{d}_1 \cdot \bar{v}_1 + \alpha_{cc1} \cdot \bar{c}_1 \cdot (\tilde{x}_2 - \bar{x}_1) + \alpha_{dc1} \cdot \bar{d}_1 \cdot (\tilde{v}_2 - \bar{v}_1) \end{aligned} \quad (a)$$

Subsystem 2:

$$\begin{aligned} \bar{x}'_2 &= \bar{v}_2 \\ \bar{v}'_2 &= -\frac{\alpha_{c21}}{\alpha_{m21}} \cdot \bar{c}_1 \cdot \bar{x}_2 - \frac{\alpha_{d21}}{\alpha_{m21}} \cdot \bar{d}_1 \cdot \bar{v}_2 \\ &\quad - \frac{\alpha_{m21}}{\alpha_{cc1}} \cdot \bar{c}_1 \cdot (\bar{x}_2 - \tilde{x}_1) - \frac{\alpha_{dc1}}{\alpha_{m21}} \cdot \bar{d}_1 \cdot (\bar{v}_2 - \tilde{v}_1) \end{aligned} \quad (b)$$

Coupling conditions:

$$\begin{aligned} \tilde{x}_1 - \bar{x}_1 &= 0 \\ \tilde{v}_1 - \bar{v}_1 &= 0 \\ \tilde{x}_2 - \bar{x}_2 &= 0 \\ \tilde{v}_2 - \bar{v}_2 &= 0. \end{aligned} \quad (c)$$

(9.22)

Both subsystems are integrated in parallel from \bar{T}_N to \bar{T}_{N+1} . For the integration of subsystem 1, the extrapolation polynomials $\tilde{x}_2(\bar{t}) = \bar{x}_2^{ex}(\bar{t})$ and $\tilde{v}_2(\bar{t}) = \bar{v}_2^{ex}(\bar{t})$ are used; subsystem 2 is integrated with the extrapolation polynomials $\tilde{x}_1(\bar{t}) = \bar{x}_1^{ex}(\bar{t})$ and $\tilde{v}_1(\bar{t}) = \bar{v}_1^{ex}(\bar{t})$, see Eqs. (9.5) and (9.6). After the subsystem integration, we have new state variables at the macro-time point \bar{T}_{N+1} similar to Eq. (9.9).

Sequential Integration (Gauss-Seidel Type). Firstly, subsystem 1 is integrated using the extrapolation polynomials $\tilde{x}_2(\bar{t}) = \bar{x}_2^{ex}(\bar{t})$ and $\tilde{v}_2(\bar{t}) = \bar{v}_2^{ex}(\bar{t})$, see Eqs. (9.5) and (9.6). Secondly, subsystem 2 is integrated from \bar{T}_N to \bar{T}_{N+1} by making use of the interpolation polynomials $\tilde{x}_1(\bar{t}) = \bar{x}_1^{in}(\bar{t})$ and $\tilde{v}_2(\bar{t}) = \bar{v}_2^{in}(\bar{t})$, see Eqs. (9.14) and (9.15). The state variables at the new macro-time point \bar{T}_{N+1} have the same structure as Eqs. (9.12) and (9.18).

9.2.4 Stability and Convergence Plots

Stability Plots. The spectral radius ρ of the system of recurrence equations, which defines the numerical stability of the underlying co-simulation method, is a nonlinear function of the 7 independent test model parameters of Eq. (9.1). To illustrate the stability behavior of co-simulation approaches in 2D-stability plots, 5 parameters are fixed and the spectral radius ρ is plotted as a function of the remaining 2 parameters. Instead of using the parameters of Eq. (9.1), it is more convenient to use the following 7 independent parameters

$$\begin{aligned}
 \bar{\Lambda}_{r1} &= -\frac{\bar{d}_1}{2}, \bar{\Lambda}_{i1} = \frac{1}{2}\sqrt{4 \cdot \bar{c}_1 - \bar{d}_1^2}, \\
 \alpha_{m21} &= \frac{m_2}{m_1}, \alpha_{\Lambda r21} = \frac{\bar{\Lambda}_{r2}}{\bar{\Lambda}_{r1}} = \frac{\alpha_{d21}}{\alpha_{m21}}, \\
 \alpha_{\Lambda i21} &= \frac{\bar{\Lambda}_{i2}}{\bar{\Lambda}_{i1}} = \frac{1}{\alpha_{m21}} \frac{\sqrt{4 \cdot \alpha_{m21} \cdot \alpha_{c21} \cdot \bar{c}_1 - \alpha_{d21}^2 \cdot \bar{d}_1^2}}{\sqrt{4 \cdot \bar{c}_1 - \bar{d}_1^2}}, \\
 \alpha_{\Lambda rc1} &= \frac{\alpha_{dc1}}{\alpha_m^*}, \alpha_{\Lambda ic1} = \frac{1}{\alpha_m^*} \frac{\sqrt{4 \cdot \alpha_m^* \cdot \alpha_{cc1} \cdot \bar{c}_1 - \alpha_{dc1}^2 \cdot \bar{d}_1^2}}{\sqrt{4 \cdot \bar{c}_1 - \bar{d}_1^2}} \\
 \text{with } \alpha_m^* &= 2 \frac{\alpha_{m21}}{1 + \alpha_{m21}}.
 \end{aligned} \tag{9.23}$$

The physical interpretation of these parameters is straightforward. $\bar{\Lambda}_{r1}$ and $\bar{\Lambda}_{i1}$ represent the (modified/dimensionless) real and imaginary part of the eigenvalue of subsystem 1. The three parameters α_{m21} , $\alpha_{\Lambda r21}$ and $\alpha_{\Lambda i21}$ characterize subsystem 2 in relation to subsystem 1. The coefficient α_{m21} describes the ratio of the subsystem masses. $\alpha_{\Lambda r21}$ and $\alpha_{\Lambda i21}$ characterize the ratio of the real and the imaginary part of the eigenvalue of subsystem 2 with respect to $\bar{\Lambda}_{r1}$ and $\bar{\Lambda}_{i1}$, i.e. the ratio of the subsystem damping and the ratio of the subsystem frequencies. To characterize the coupling of the subsystems, the two parameters $\alpha_{\Lambda rc1}$ and $\alpha_{\Lambda ic1}$ are introduced. They may be interpreted as follows. For $c_1 = c_2 = d_1 = d_2 = 0$, the oscillation of the two masses is characterized by the eigenvalue $-\frac{\alpha_{dc1} \cdot \bar{d}_1}{\alpha_m^*} + i \cdot \frac{\sqrt{2 \cdot \alpha_m^* \cdot \alpha_{cc1} \cdot \bar{c}_1 - \alpha_{dc1}^2 \cdot \bar{d}_1^2}}{\alpha_m^*}$. The two parameters $\alpha_{\Lambda rc1}$ and $\alpha_{\Lambda ic1}$ are closely related with the ratio of the real and imaginary part of this eigenvalue with respect to $\bar{\Lambda}_{r1}$ and $\bar{\Lambda}_{i1}$. However, a factor 2 has artificially been introduced so that for the symmetric case ($m_1 = m_2$, $c_1 = c_2 = c_c$ and $d_1 = d_2 = d_c$) the parameters α_{m21} , $\alpha_{\Lambda r21}$, $\alpha_{\Lambda i21}$, $\alpha_{\Lambda rc1}$ and $\alpha_{\Lambda ic1}$ become 1. In other words, $\alpha_{\Lambda rc1}$ and $\alpha_{\Lambda ic1}$ characterize the ratio of the real and imaginary part of the Eigenvalue of the unfixed oscillator ($c_1 = d_1 = c_2 = d_2 = 0$) with respect to subsystem 1.

Considering time-integration schemes, only two parameters are required to fully describe the numerical stability behaviour, namely $Re\{h\lambda\}$ and $Im\{h\lambda\}$ (λ is the Eigenvalue of Dahlquist's test equation and h the integration step size). For co-simulation

methods, 7 parameters are required to define the test model and to characterize the numerical stability, since the co-simulation test model is represented by two coupled Dahlquist equations. Consequently, a complete illustration of the numerical stability with 2D-plots is not possible for co-simulation methods, since 7-dimensional graphs would be necessary for a complete representation of the numerical stability. It seems therefore to be reasonable to use the two parameters $\bar{\Lambda}_{r1}$ and $\bar{\Lambda}_{i1}$, which fully define subsystem 1, as basis parameters to be varied in 2D-stability plots. The remaining 5 parameters describe subsystem 2 ($\alpha_{m21}, \alpha_{\Lambda r21}, \alpha_{\Lambda i21}$) and the coupling element ($\alpha_{\Lambda rc1}, \alpha_{\Lambda ic1}$) in relation to subsystem 1. It should be stressed that the choice of the two parameters to be varied in 2D-stability plots is arbitrary. Instead of $\bar{\Lambda}_{r1}$ and $\bar{\Lambda}_{i1}$, two other of the altogether 7 parameters might be chosen as axes for 2D-stability plots, which—as a matter of course—would yield different stability plots.

In the following subsections, 2D-stability plots are presented for the symmetric test model. Therefore, the parameters $\bar{\Lambda}_{r1}$ and $\bar{\Lambda}_{i1}$ are varied in the range $[-1, 0]$ and $[0, 2]$. It should be stressed that the spectral radius ρ can only be computed numerically. To reduce floating point errors, computation of ρ has been accomplished very precisely by using 128 digit points. Note that every dot/circle indicates a stable point, that is a point for which the spectral radius ρ of the governing system of recurrence equations is less or equal to 1 (i.e. a point for which $\rho \leq (1 + 10^{-10})$ holds). The color of the dots just reflects the magnitude of the spectral radius. Since the spectral radius can only be calculated numerically, one has to fix all 7 parameters in order to calculate ρ . That implies that the region of stability (instability) can only be determined by checking discrete points. Strictly speaking, it could theoretically be possible that there exist unstable points “between” two stable points (i.e. between two dots), since the discretization of the grid is finite. In our numerical tests with a finer discretization, we did, however, not observe such a behavior.

Stability plots for the integrated acceleration co-simulation approaches are compared with corresponding plots for the classical co-simulation approaches [71] in order to show the improved stability behavior of the new coupling techniques. It should be mentioned that the notion “classical” is used to denote the well-established co-simulation approaches, where the position and velocity variables are extrapolated by Lagrange polynomials of arbitrary order. The integrated acceleration approach shows a quadratic convergence behavior for the global error already for the case of constant approximation polynomials ($k = 0$), see Fig. 9.11. To obtain quadratic convergence, linear polynomials have to be used in case of the classical approach, see Ref. [71]. Hence, in order to compare the integrated acceleration approach with the classical approach, linear Lagrange polynomials have to be used for the classical method.

Remark: In this paper, a stability analysis is presented for explicit co-simulation methods on the basis of two coupled Dahlquist equations. According to the stability analysis for time integration schemes, we ride on the assumption that the macro-step size is constant. Due to the fact that the test model is linear, the subsystem integration can be carried out analytically. Hence, stabilizing or destabilizing effects due to a numerical subsystem integration are not present in our analysis and a multirate factor

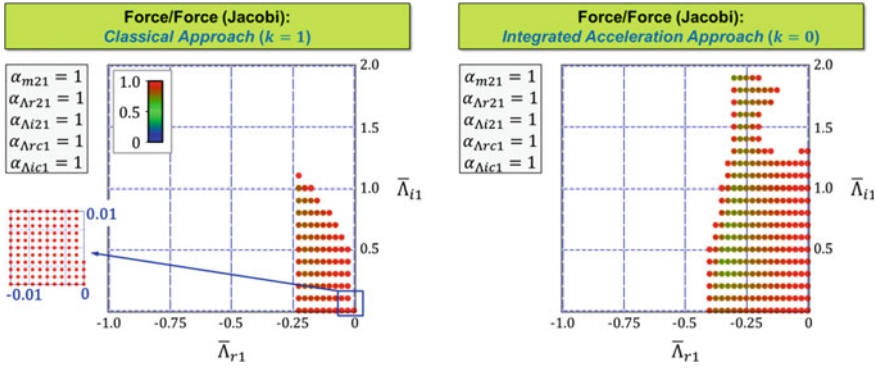


Fig. 9.5 Stability plots (Jacobi type, force/force-decomposition): classical approach ($k = 1$) and integrated acceleration approach ($k = 0$)

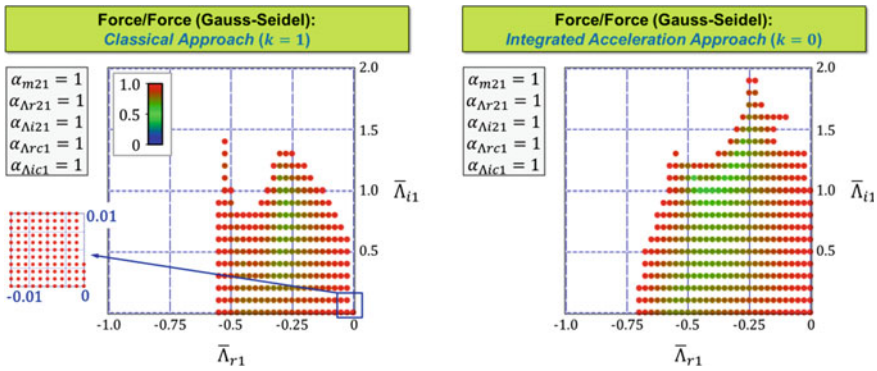


Fig. 9.6 Stability plots (Gauss-Seidel type, force/force-decomposition): classical approach ($k = 1$) and integrated acceleration approach ($k = 0$)

has not to be taken into account [36, 39, 45]. The results only reflect numerical instabilities introduced by the co-simulation, i.e. by the weak coupling approach.

Force/Force-Coupling. Stability plots for the force/force-decomposition approach in connection with the Jacobi integration type are shown in Fig. 9.5. The left plot represents the stability behavior of the classical explicit co-simulation approach based on linear Lagrange polynomials ($k = 1$), whereas the right plot characterizes the numerical stability of the integrated acceleration approach based on constant approximation ($k = 0$). Corresponding plots for the sequential Gauss-Seidel integration type are collected in Fig. 9.6.

From the plots, we can firstly observe that the region of stability is markedly larger for the integrated acceleration co-simulation approach. Of special interest is the highly improved stability behavior for undamped systems, represented by the stable points on the $\bar{\Lambda}_{i1}$ -axes. Note that $\bar{\Lambda}_{r1} = -\frac{\bar{d}_1}{2} = 0$ implicates that $\bar{d}_1 =$

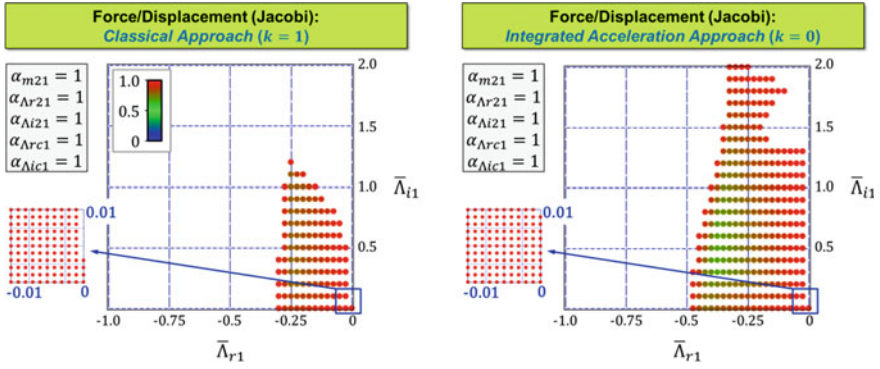


Fig. 9.7 Stability plots (Jacobi type, force/displacement-decomposition): classical approach ($k = 1$) and integrated acceleration approach ($k = 0$)

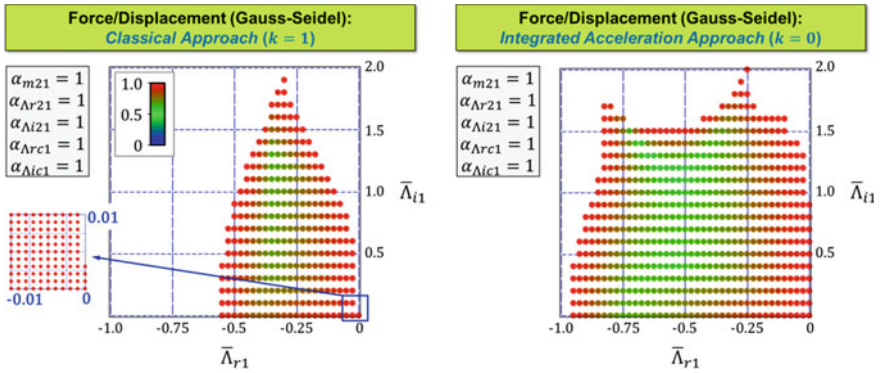


Fig. 9.8 Stability plots (Gauss-Seidel type, force/displacement-decomposition): classical approach ($k = 1$) and integrated acceleration approach ($k = 0$)

$\frac{d_1 \cdot H}{m_1} = 0$ and therefore that $d_1 = 0$. With $\alpha_{\Lambda r21} = \frac{\bar{\Lambda}_{r2}}{\bar{\Lambda}_{r1}} = \frac{\alpha_{d21}}{\alpha_{m21}}$ and $\alpha_{d21} = \frac{d_2}{d_1}$, we get $d_2 = \alpha_{\Lambda r21} \cdot \alpha_{m21} \cdot d_1 = 0$, which is zero for $d_1 = 0$. The stable points on the $\bar{\Lambda}_{i1}$ -axes are important from the practical point of view, since the integrated acceleration approach offers the opportunity to couple undamped systems by an explicit co-simulation technique with comparatively large macro-step sizes.

Force/Displacement-Coupling. Stability plots for the force/displacement-decomposition approach are collected in Figs. 9.7 and 9.8. As for the force/force-coupling approach, we observe a significantly improved stability behavior for the integrated acceleration approach.

Displacement/Displacement-Coupling. The numerical stability of the classical and of the integrated acceleration co-simulation approach is illustrated in Figs. 9.9 and 9.10 for the case of displacement/displacement-decomposition. Again, we detect that the region of stability is larger for the integrated acceleration approach.

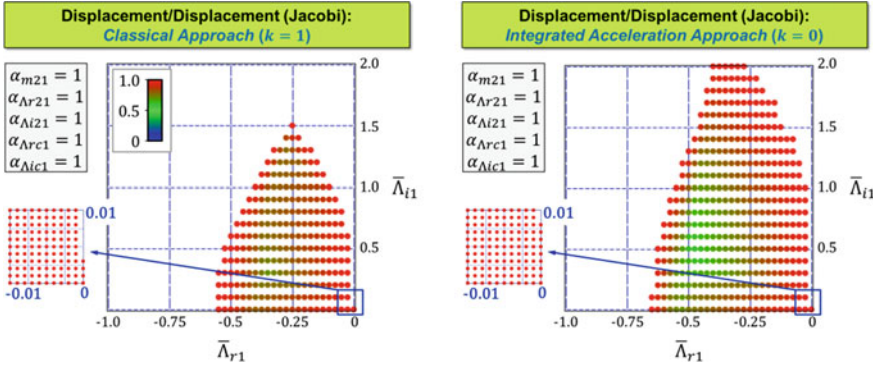


Fig. 9.9 Stability plots (Jacobi type, displacement/displacement-decomposition): classical approach ($k = 1$) and integrated acceleration approach ($k = 0$)

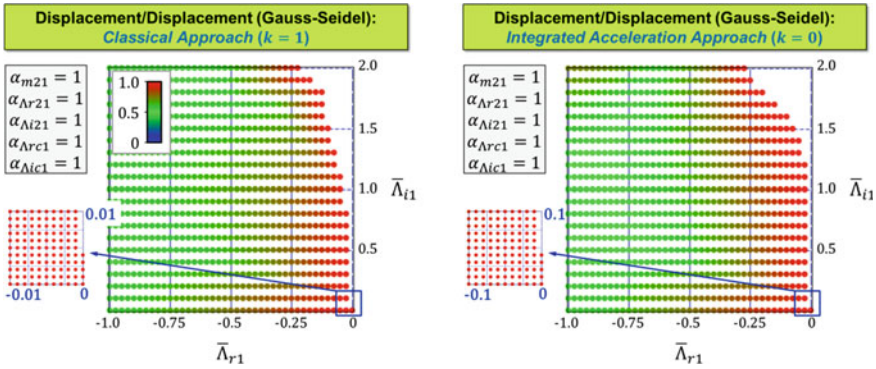


Fig. 9.10 Stability plots (Gauss-Seidel type, displacement/displacement-decomposition): classical approach ($k = 1$) and integrated acceleration approach ($k = 0$)

Convergence Plots. Next, the convergence behavior of the integrated acceleration co-simulation approach is analyzed. Therefore, simulations have been carried out with the symmetrical co-simulation test model using the following parameters: $m_1 = m_2 = 1, c_1 = c_2 = c_c = 1000, d_1 = d_2 = d_c = 10$. The relative global error of the position variables x_1, x_2 is computed with the help of the normalized root mean square error (NRMSE) according to

$$\varepsilon_{pos, glo} = \sqrt{\frac{\sum_N (x_{1, glo}(T_N) - x_{1, N})^2}{\sum_N (x_{1, glo}(T_N) - x_{1, mean})^2} + \frac{\sum_N (x_{2, glo}(T_N) - x_{2, N})^2}{\sum_N (x_{2, glo}(T_N) - x_{2, mean})^2}}$$

with $x_{1, glo}(T_N) = \int_{T_0}^{T_N} v_1(t) dt + x_1(T_0)$

$$\begin{aligned}
\text{and } x_{2,glo}(T_N) &= \int_{T_0}^{T_N} v_2(t)dt + x_2(T_0), \\
\text{and } x_{1,mean} &= \sum_N \frac{x_{1,glo}(T_N)}{N_{total}} \quad \text{and} \quad x_{2,mean} = \sum_N \frac{x_{2,glo}(T_N)}{N_{total}}. \quad (9.24)
\end{aligned}$$

In the above equation, the values $x_{1,N}, x_{2,N}$ denote the co-simulation results at the macro-time point T_N (solution of the governing recurrence system) and $x_1(T_N), x_2(T_N)$ the values of the analytical solution, which are integrated analytically from T_0 to T_N . N_{total} represents the total number of macro-steps. The global error $\varepsilon_{vel,glo}$ of the velocity variables v_1, v_2 and the related local errors $\varepsilon_{pos,loc}$ and $\varepsilon_{vel,loc}$ of the position and velocity variables are computed in a similar manner. For instance, the local error $\varepsilon_{pos,loc}$ is computed by

$$\begin{aligned}
\varepsilon_{pos,loc} &= \sqrt{\frac{\sum_N (x_{1,loc}(T_N) - x_{1,N})^2}{\sum_N (x_{1,loc}(T_N) - x_{1,mean})^2} + \frac{\sum_N (x_{2,loc}(T_N) - x_{2,N})^2}{\sum_N (x_{2,loc}(T_N) - x_{2,mean})^2}} \\
\text{with } x_{1,loc}(T_N) &= \int_{T_{N-1}}^{T_N} v_1(t)dt + x_{1,N-1} \\
\text{with } x_{2,loc}(T_N) &= \int_{T_{N-1}}^{T_N} v_2(t)dt + x_{2,N-1} \\
\text{and } x_{1,mean} &= \sum_N \frac{x_{1,loc}(T_N)}{N_{total}} \quad \text{and} \quad x_{2,mean} = \sum_N \frac{x_{2,loc}(T_N)}{N_{total}}. \quad (9.25)
\end{aligned}$$

The only difference to the global error definition is that the values of the analytical solutions $x_{1,loc}(T_N), x_{2,loc}(T_N)$ are integrated analytically from T_{N-1} to T_N using as initial conditions the co-simulation results $x_{1,N-1}, x_{2,N-1}$ from the previous macro-step T_{N-1} .

Figure 9.11 shows the errors $\varepsilon_{pos,glo}, \varepsilon_{vel,glo}, \varepsilon_{pos,loc}$ and $\varepsilon_{vel,loc}$ of the integrated acceleration co-simulation approach ($k = 0$) for both the parallel Jacobi-type and the sequential Gauss-Seidel type and for all three decomposition techniques. Regarding the convergence plots for the classical approach, see Fig. 12 in Ref. [71], we observe that the integrated acceleration approach for $k = 0$ exhibits the same convergence behavior as the classical approach for $k = 1$. The error magnitudes are, however, notably smaller for the integrated acceleration approach. As can be seen, the global errors $\varepsilon_{pos,glo}$ and $\varepsilon_{vel,glo}$ converge with $\mathcal{O}(H^2)$; the local errors $\varepsilon_{pos,loc}$ and $\varepsilon_{vel,loc}$ converge with $\mathcal{O}(H^4)$ and $\mathcal{O}(H^3)$, respectively. It should be stressed that the convergence plots of Fig. 9.11 have been generated with the linear test model. Due to the linearity of the two-mass oscillator, the subsystem integration can be carried out analytically. Consequently, the plots are not influenced by any numerical errors introduced by a numerical time integration of the subsystems and therefore accurately reflect the convergence behavior of the coupling scheme. In general, the convergence behavior of a co-simulation may also be influenced by the numerical subsystem integration, especially for the case that the micro-step size for the subsystem integration is simi-

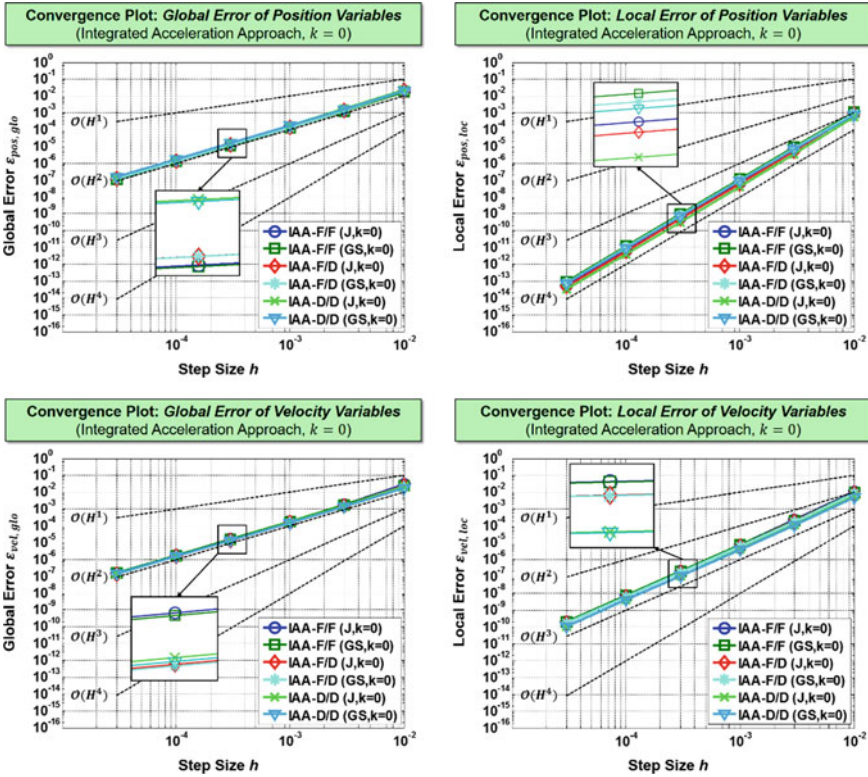


Fig. 9.11 Convergence plots for the integrated acceleration co-simulation approach ($k = 0$): global and local error of the position and velocity variables over the macro-step size H

lar to the macro-step size of the co-simulation. If the macro-step size is significantly larger than the micro-step size, the numerical error introduced by the co-simulation usually dominates the overall error [7].

Convergence plots for the integrated acceleration approach based on linear approximation ($k = 1$) can be found in Appendix 1.

9.3 Generalizing for Coupling Arbitrary Mechanical Systems

Next, the integrated acceleration co-simulation approach introduced in Sect. 9.2 is generalized for the case that two multibody systems are coupled by arbitrary nonlinear constitutive equations.

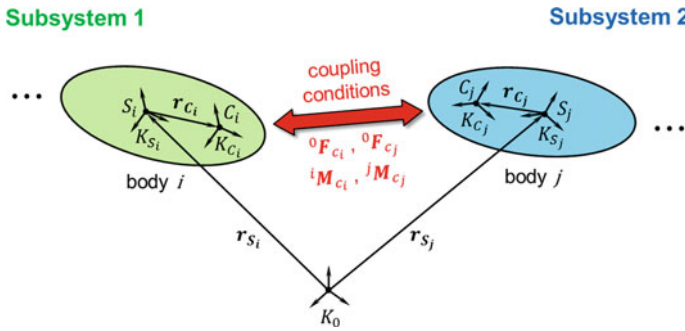


Fig. 9.12 Body i of subsystem 1 coupled with body j of subsystem 2 by constitutive equations (coupling points C_i and C_j)

9.3.1 Definition of the Two Mechanical Subsystems

We consider two mechanical subsystems, which are both assumed to be general multibody systems. In this manuscript, absolute coordinates are used in order to formulate the equations of motion [64, 66, 78]. It should be mentioned that finite-element systems may be treated in a very similar manner. Subsystem 1 consists of n_1 rigid bodies, while subsystem 2 contains n_2 rigid bodies. It is assumed that body i of subsystem 1 is coupled with body j of subsystem 2 by an arbitrary nonlinear bushing element, which is in general described by 12 nonlinear constitutive equations (coupling conditions). The bushing element is connected to body i at the coupling point C_i and to body j at the coupling point C_j , see Fig. 9.12. The center of mass S_i of coupling body i (mass m_i , inertia tensor iJ_i with respect to the body fixed principal axes system K_{S_i}) is defined by the position vector r_{S_i} . The coupling point C_i is specified by the body fixed vector r_{C_i} . For subsystem 2, an equivalent notation is used. The inertial reference frame is denoted by K_0 .

Let ${}^{nm}T$ denote the transformation matrix, which transforms the coordinates of an arbitrary vector a from system K_m to system K_n by means of the relationship ${}^na = {}^{nm}T^ma$. It should be noted that the left upper index specifies the coordinate system to be used for decomposing the vector. Position and velocity of an arbitrary rigid body q (center of mass S_q) are described by the vector coordinates ${}^0r_{S_q}$ and ${}^0v_{S_q} = {}^0\dot{r}_{S_q}$. To describe the rotation of the rigid body q , three rotation parameters are used to be collected in the vector $\gamma_q \in \mathbb{R}^3$. The angular velocity of body q , decomposed with respect to the body fixed system K_{S_q} , is denoted by ${}^q\omega_q$. The time derivative of the rotation parameters are related to the angular velocity by means of the kinematical differential equation $\dot{\gamma}_q = B(\gamma_q)q\omega_q$, where B represents a (3×3) -matrix. For instance, using Bryant angles $\gamma_q = (\alpha_q \beta_q \gamma_q)^T$ as rotation parameters, the matrix $B(\gamma_q) \in \mathbb{R}^{3 \times 3}$ is given by

$$\mathbf{B} = \frac{1}{\cos \beta_q} \begin{pmatrix} \cos \gamma_q & -\sin \gamma_q & 0 \\ \cos \beta_q \sin \gamma_q & \cos \beta_q \cos \gamma_q & 0 \\ -\sin \beta_q \cos \gamma_q & \sin \beta_q \sin \gamma_q & \cos \beta_q \end{pmatrix}. \quad (9.26)$$

For the following analysis, it is useful to introduce some auxiliary vectors. The two vectors $\mathbf{z}_i = (z_{i1} \ z_{i2} \ \dots \ z_{i12})^T = ({}^0\mathbf{r}_{S_i} \ \boldsymbol{\gamma}_i \ {}^0\mathbf{v}_{S_i} \ {}^i\boldsymbol{\omega}_i)^T \in \mathbb{R}^{12}$ and $\mathbf{z}_j = (z_{j1} \ z_{j2} \ \dots \ z_{j12})^T = ({}^0\mathbf{r}_{S_j} \ \boldsymbol{\gamma}_j \ {}^0\mathbf{v}_{S_j} \ {}^j\boldsymbol{\omega}_j)^T \in \mathbb{R}^{12}$ collect the position and velocity coordinates of the coupling bodies i and j . They are combined in the vector $\mathbf{z}_c = (\mathbf{z}_i \ \mathbf{z}_j)^T \in \mathbb{R}^{24}$. The position and velocity coordinates of all bodies of subsystem 1 are collected in the vector $\hat{\mathbf{z}}_1 \in \mathbb{R}^{12 \cdot n_1}$. Accordingly, $\hat{\mathbf{z}}_2 \in \mathbb{R}^{12 \cdot n_2}$ contains all state variables of subsystem 2.

The equations of motion for the coupling bodies i and j , i.e. the Newton-Euler equations, read as

Coupling body i (subsystem 1):

$$\begin{aligned} {}^0\dot{\mathbf{r}}_{S_i} &= {}^0\mathbf{v}_{S_i} & (a) \\ m_i {}^0\dot{\mathbf{v}}_{S_i} &= {}^0\mathbf{F}_{a_i}(\hat{\mathbf{z}}_1, t) + {}^0\mathbf{F}_{r_i} + {}^0\mathbf{F}_{c_i} & (b) \\ \dot{\boldsymbol{\gamma}}_i &= \mathbf{B}(\boldsymbol{\gamma}_i) {}^i\boldsymbol{\omega}_i & (c) \\ {}^i\mathbf{J}_i^i \dot{\boldsymbol{\omega}}_i + {}^i\boldsymbol{\omega}_i \times {}^i\mathbf{J}_i^i \boldsymbol{\omega}_i &= {}^i\mathbf{M}_{a_i}(\hat{\mathbf{z}}_1, t) + {}^i\mathbf{M}_{r_i} + {}^i\mathbf{r}_{C_i} \times {}^i\mathbf{F}_{c_i} + {}^i\mathbf{M}_{c_i} & (d) \end{aligned} \quad (9.27)$$

Coupling body j (subsystem 2):

$$\begin{aligned} {}^0\dot{\mathbf{r}}_{S_j} &= {}^0\mathbf{v}_{S_j} & (a) \\ m_j {}^0\dot{\mathbf{v}}_{S_j} &= {}^0\mathbf{F}_{a_j}(\hat{\mathbf{z}}_2, t) + {}^0\mathbf{F}_{r_j} + {}^0\mathbf{F}_{c_j} & (b) \\ \dot{\boldsymbol{\gamma}}_j &= \mathbf{B}(\boldsymbol{\gamma}_j) {}^j\boldsymbol{\omega}_j & (c) \\ {}^j\mathbf{J}_j^j \dot{\boldsymbol{\omega}}_j + {}^j\boldsymbol{\omega}_j \times {}^j\mathbf{J}_j^j \boldsymbol{\omega}_j &= {}^j\mathbf{M}_{a_j}(\hat{\mathbf{z}}_2, t) + {}^j\mathbf{M}_{r_j} + {}^j\mathbf{r}_{C_j} \times {}^j\mathbf{F}_{c_j} + {}^j\mathbf{M}_{c_j} & (d) \end{aligned} \quad (9.28)$$

In the above equations, the vectors ${}^0\mathbf{F}_{a_i}$ and ${}^i\mathbf{M}_{a_i}$ denote applied forces and torques acting on coupling body i . The reaction forces and torques, originating from algebraic constraint equations in subsystem 1, are indicated by ${}^0\mathbf{F}_{r_i}$ and ${}^i\mathbf{M}_{r_i}$. The coupling forces and torques, resulting from the bushing element between body i and body j , are represented by ${}^0\mathbf{F}_{c_i}$ and ${}^i\mathbf{M}_{c_i}$. For subsystem 2, an equivalent notation is used.

9.3.2 Coupling Equations Between the Subsystems

The two subsystems are assumed to be coupled by an arbitrary nonlinear bushing element, which is mathematically defined by a set of 12 nonlinear scalar constitutive equations. The constitutive equations represent a nonlinear relationship between the coupling forces/torques and the position/velocity variables of the two coupling bodies, i.e. equations of the form

$$\begin{aligned} {}^0\mathbf{F}_{C_i} &= {}^0\mathbf{F}_{C_i}(\tilde{\mathbf{z}}_i, \tilde{\mathbf{z}}_j), \quad {}^i\mathbf{M}_{C_i} = {}^i\mathbf{M}_{C_i}(\tilde{\mathbf{z}}_i, \tilde{\mathbf{z}}_j), \\ {}^0\mathbf{F}_{C_j} &= {}^0\mathbf{F}_{C_j}(\tilde{\mathbf{z}}_i, \tilde{\mathbf{z}}_j), \quad {}^j\mathbf{M}_{C_j} = {}^j\mathbf{M}_{C_j}(\tilde{\mathbf{z}}_i, \tilde{\mathbf{z}}_j) \end{aligned} \quad (9.29)$$

with $\tilde{\mathbf{z}}_i = ({}^0\mathbf{r}_{S_i} \ \boldsymbol{\gamma}_i \ {}^0\mathbf{v}_{S_i} \ \dot{\boldsymbol{\gamma}}_i)^T \in \mathbb{R}^{12}$ and $\tilde{\mathbf{z}}_j = ({}^0\mathbf{r}_{S_j} \ \boldsymbol{\gamma}_j \ {}^0\mathbf{v}_{S_j} \ \dot{\boldsymbol{\gamma}}_j)^T \in \mathbb{R}^{12}$.

In the general 3D case, the coupling forces fulfill $\mathbf{F}_{C_i} = -\mathbf{F}_{C_j}$ (action = reaction). For the torques, however, we usually have $\mathbf{M}_{C_i} \neq -\mathbf{M}_{C_j}$.

Example Let $\mathbf{e}_{x_i}, \mathbf{e}_{y_i}, \mathbf{e}_{z_i}$ and $\mathbf{e}_{x_j}, \mathbf{e}_{y_j}, \mathbf{e}_{z_j}$ denote the unit-vectors of the body fixed systems K_{C_i} and K_{C_j} at the coupling points C_i and C_j , see Fig. 9.12. Assuming small relative rotations (i.e. stiff rotational coupling), the relative rotation angles (projection angles) of system K_{C_i} with respect to system K_{C_j} —collected in the vector $\Delta\boldsymbol{\alpha} = (\Delta\alpha_x \ \Delta\alpha_y \ \Delta\alpha_z)^T$ —can be calculated by means of

$$\begin{aligned} \Delta\alpha_x &= \text{atan2}(-\mathbf{e}_{z_i} \cdot \mathbf{e}_{y_j}, \mathbf{e}_{z_i} \cdot \mathbf{e}_{z_j}), \\ \Delta\alpha_y &= \text{atan2}(\mathbf{e}_{z_i} \cdot \mathbf{e}_{x_j}, \mathbf{e}_{z_i} \cdot \mathbf{e}_{z_j}), \\ \Delta\alpha_z &= \text{atan2}(\mathbf{e}_{x_i} \cdot \mathbf{e}_{y_j}, \mathbf{e}_{x_i} \cdot \mathbf{e}_{x_j}). \end{aligned} \quad (9.30)$$

Note that the relative rotation angles can be expressed as functions of the Bryant angles of the two coupling bodies, i.e. $\Delta\alpha_x = \Delta\alpha_x(\boldsymbol{\gamma}_i, \boldsymbol{\gamma}_j)$, $\Delta\alpha_y = \Delta\alpha_y(\boldsymbol{\gamma}_i, \boldsymbol{\gamma}_j)$, $\Delta\alpha_z = \Delta\alpha_z(\boldsymbol{\gamma}_i, \boldsymbol{\gamma}_j)$. Connecting the coupling points C_i and C_j by a 3D linear spring/damper system (3D linear bushing element), the constitutive equations read as

$$\begin{aligned} {}^0\mathbf{F}_{C_j} &= -{}^0\mathbf{F}_{C_i} = \mathbf{C}_T \cdot [({}^0\mathbf{r}_{S_i} + {}^{0i}\mathbf{T}(\boldsymbol{\gamma}_i) {}^i\mathbf{r}_{C_i}) - ({}^0\mathbf{r}_{S_j} + {}^{0j}\mathbf{T}(\boldsymbol{\gamma}_j) {}^j\mathbf{r}_{C_j})] \\ &\quad + \mathbf{D}_T \cdot [({}^0\mathbf{v}_{S_i} + {}^{0i}\mathbf{T}(\boldsymbol{\gamma}_i) {}^i\boldsymbol{\omega}_i \times {}^{0i}\mathbf{T}(\boldsymbol{\gamma}_i) {}^i\mathbf{r}_{C_i}) \\ &\quad - ({}^0\mathbf{v}_{S_j} + {}^{0j}\mathbf{T}(\boldsymbol{\gamma}_j) {}^j\boldsymbol{\omega}_j \times {}^{0j}\mathbf{T}(\boldsymbol{\gamma}_j) {}^j\mathbf{r}_{C_j})], \\ {}^j\mathbf{M}_{C_j} &= \mathbf{C}_R \cdot \Delta\boldsymbol{\alpha} + \mathbf{D}_R \cdot ({}^{ji}\mathbf{T}(\boldsymbol{\gamma}_i, \boldsymbol{\gamma}_j) {}^i\boldsymbol{\omega}_i - {}^j\boldsymbol{\omega}_j), \\ {}^i\mathbf{M}_{C_i} &= -{}^{ij}\mathbf{T}(\boldsymbol{\gamma}_i, \boldsymbol{\gamma}_j) {}^j\mathbf{M}_{C_j} \\ &\quad - {}^{i0}\mathbf{T}(\boldsymbol{\gamma}_j) \{ [({}^0\mathbf{r}_{S_j} + {}^{0j}\mathbf{T}(\boldsymbol{\gamma}_j) {}^j\mathbf{r}_{C_j}) - ({}^0\mathbf{r}_{S_i} + {}^{0i}\mathbf{T}(\boldsymbol{\gamma}_i) {}^i\mathbf{r}_{C_i})] \times {}^0\mathbf{F}_{C_j} \}, \end{aligned} \quad (9.31)$$

where the diagonal matrices $\mathbf{C}_T = \text{diag}(c_{cx}, c_{cy}, c_{cz})$ and $\mathbf{C}_R = \text{diag}(c_{c\alpha_x}, c_{c\alpha_y}, c_{c\alpha_z})$ contain the three translational and rotational spring constants and the diagonal matrices $\mathbf{D}_T = \text{diag}(d_x, d_y, d_z)$ and $\mathbf{D}_R = \text{diag}(d_{c\alpha_x}, d_{c\alpha_y}, d_{c\alpha_z})$ the three translational and rotational damping coefficients (non-diagonal elements are neglected for the reason of a clear representation).

9.3.3 Co-simulation Algorithm

For the reason of a concise representation, only the case force/force-decomposition is discussed next. Force/displacement- and displacement/displacement-decomposition can be treated in a very similar manner and will therefore not be presented in detail here.

Force/Force-Coupling Approach Using Parallel Integration (Jacobi Type).

At the beginning of the macro-step, the state variables of the two subsystems are assumed to be known, namely

$$\hat{\mathbf{z}}_1(t = T_N) = \hat{\mathbf{z}}_{1,N}, \hat{\mathbf{z}}_2(t = T_N) = \hat{\mathbf{z}}_{2,N}. \quad (9.32)$$

Step 1: Integration of Extrapolated Accelerations

- With the state variables $\hat{\mathbf{z}}_{1,N}$ and $\hat{\mathbf{z}}_{2,N}$ and with the help of the equations of motion, the accelerations ${}^0\dot{\mathbf{v}}_{S_i,N}$, ${}^0\ddot{\mathbf{y}}_{i,N}$, ${}^0\dot{\mathbf{v}}_{S_j,N}$, ${}^0\ddot{\mathbf{y}}_{j,N}$ of the two coupling bodies can be calculated. Using the accelerations at the macro-time points $T_N, T_{N-1}, \dots, T_{N-k}$, extrapolation polynomials ${}^0\dot{\mathbf{v}}_{S_i}^{\text{ex}}(t)$, ${}^0\ddot{\mathbf{y}}_i^{\text{ex}}(t)$, ${}^0\dot{\mathbf{v}}_{S_j}^{\text{ex}}(t)$, ${}^0\ddot{\mathbf{y}}_j^{\text{ex}}(t)$ of degree k can be generated for the time interval $[T_N, T_{N+1}]$.
- An analytical forward integration of the polynomials ${}^0\dot{\mathbf{v}}_{S_i}^{\text{ex}}(t)$, ${}^0\ddot{\mathbf{y}}_i^{\text{ex}}(t)$, ${}^0\dot{\mathbf{v}}_{S_j}^{\text{ex}}(t)$, ${}^0\ddot{\mathbf{y}}_j^{\text{ex}}(t)$ from T_N to T_{N+1} with the initial conditions ${}^0\mathbf{v}_{S_i,N}$, ${}^0\dot{\mathbf{y}}_{i,N}$, ${}^0\mathbf{r}_{S_i,N}$, $\boldsymbol{\gamma}_{i,N}$ and ${}^0\mathbf{v}_{S_j,N}$, ${}^0\dot{\mathbf{y}}_{j,N}$, ${}^0\mathbf{r}_{S_j,N}$, $\boldsymbol{\gamma}_{j,N}$ yields extrapolated state variables for the time interval $[T_N, T_{N+1}]$

$$\begin{aligned} &{}^0\mathbf{v}_{S_i}^{\text{ex}}(t), {}^0\dot{\mathbf{y}}_i^{\text{ex}}(t), {}^0\mathbf{r}_{S_i}^{\text{ex}}(t), \boldsymbol{\gamma}_i^{\text{ex}}(t), \\ &{}^0\mathbf{v}_{S_j}^{\text{ex}}(t), {}^0\dot{\mathbf{y}}_j^{\text{ex}}(t), {}^0\mathbf{r}_{S_j}^{\text{ex}}(t), \boldsymbol{\gamma}_j^{\text{ex}}(t). \end{aligned} \quad (9.33)$$

- Note that for the special case of constant approximation ($k = 0$), the subsequent variables are obtained

$$\begin{aligned} &{}^0\mathbf{v}_{S_i}^{\text{ex}}(t) = {}^0\dot{\mathbf{v}}_{S_i,N} \cdot (t - T_N) + {}^0\mathbf{v}_{S_i,N}, \quad \dot{\mathbf{y}}_i^{\text{ex}}(t) = \ddot{\mathbf{y}}_{i,N} \cdot (t - T_N) + \dot{\mathbf{y}}_{i,N}, \\ &{}^0\mathbf{r}_{S_i}^{\text{ex}}(t) = \frac{1}{2} {}^0\dot{\mathbf{v}}_{S_i,N} \cdot (t - T_N)^2 \quad \boldsymbol{\gamma}_i^{\text{ex}}(t) = \frac{1}{2} \ddot{\mathbf{y}}_{i,N} \cdot (t - T_N)^2 \\ &\quad + {}^0\mathbf{v}_{S_i,N} \cdot (t - T_N) + {}^0\mathbf{r}_{S_i,N}, \quad + \dot{\mathbf{y}}_{i,N} \cdot (t - T_N) + \boldsymbol{\gamma}_{i,N} \\ &{}^0\mathbf{v}_{S_j}^{\text{ex}}(t) = {}^0\dot{\mathbf{v}}_{S_j,N} \cdot (t - T_N) + {}^0\mathbf{v}_{S_j,N}, \quad \dot{\mathbf{y}}_j^{\text{ex}}(t) = \ddot{\mathbf{y}}_{j,N} \cdot (t - T_N) + \dot{\mathbf{y}}_{j,N}, \\ &{}^0\mathbf{r}_{S_j}^{\text{ex}}(t) = \frac{1}{2} {}^0\dot{\mathbf{v}}_{S_j,N} \cdot (t - T_N)^2 \quad \boldsymbol{\gamma}_j^{\text{ex}}(t) = \frac{1}{2} \ddot{\mathbf{y}}_{j,N} \cdot (t - T_N)^2 \\ &\quad + {}^0\mathbf{v}_{S_j,N} \cdot (t - T_N) + {}^0\mathbf{r}_{S_j,N} \quad + \dot{\mathbf{y}}_{j,N} \cdot (t - T_N) + \boldsymbol{\gamma}_{j,N} \end{aligned} \quad (9.34)$$

Step 2: Calculation of Extrapolated Coupling Forces/Torques

- Inserting the extrapolated state variables of Eq. (9.33) into the constitutive Eqs. (9.29), extrapolation polynomials of degree $k + 2$ for the coupling forces/torques for the time interval $[T_N, T_{N+1}]$ can be calculated, namely

$$\begin{aligned} {}^0\mathbf{F}_{c_i}^{ex}(t) &= {}^0\mathbf{F}_{c_i}^{ex}(\tilde{\mathbf{z}}_i^{ex}(t), \tilde{\mathbf{z}}_j^{ex}(t)), {}^i\mathbf{M}_{c_i}^{ex}(t) = {}^i\mathbf{M}_{c_i}^{ex}(\tilde{\mathbf{z}}_i^{ex}(t), \tilde{\mathbf{z}}_j^{ex}(t)), \\ {}^0\mathbf{F}_{c_j}^{ex}(t) &= {}^0\mathbf{F}_{c_j}^{ex}(\tilde{\mathbf{z}}_i^{ex}(t), \tilde{\mathbf{z}}_j^{ex}(t)), {}^j\mathbf{M}_{c_j}^{ex}(t) = {}^j\mathbf{M}_{c_j}^{ex}(\tilde{\mathbf{z}}_i^{ex}(t), \tilde{\mathbf{z}}_j^{ex}(t)) \end{aligned} \quad (9.35)$$

with $\tilde{\mathbf{z}}_i^{ex}(t) = \left({}^0\mathbf{r}_{S_i}^{ex}(t) \ \boldsymbol{\gamma}_i^{ex}(t) \ {}^0\mathbf{v}_{S_i}^{ex}(t) \ \dot{\boldsymbol{\gamma}}_i^{ex}(t) \right)^T \in \mathbb{R}^{12}$ and

$$\tilde{\mathbf{z}}_j^{ex}(t) = \left({}^0\mathbf{r}_{S_j}^{ex}(t) \ \boldsymbol{\gamma}_j^{ex}(t) \ {}^0\mathbf{v}_{S_j}^{ex}(t) \ \dot{\boldsymbol{\gamma}}_j^{ex}(t) \right)^T \in \mathbb{R}^{12}.$$

Step 3: Integration of Subsystems

- Integrating subsystem 1 and 2 from T_N to T_{N+1} with the initial conditions (9.32) and with the extrapolated coupling forces/torques of Eq. (9.35), we obtain the new state variables at the macro-time point T_{N+1} , that is

$$\hat{\mathbf{z}}_{1,N+1}, \quad \hat{\mathbf{z}}_{2,N+1}.$$

- Finally, the accelerations ${}^0\dot{\mathbf{v}}_{S_i,N+1}$, $\ddot{\boldsymbol{\gamma}}_{i,N+1}$, ${}^0\dot{\mathbf{v}}_{S_j,N+1}$, $\ddot{\boldsymbol{\gamma}}_{j,N+1}$ of subsystem 1 and subsystem 2 are updated with the help of the equations of motion by using the coupling forces/torques ${}^0\mathbf{F}_{c_i}(\tilde{\mathbf{z}}_{i,N+1}, \tilde{\mathbf{z}}_{j,N+1})$, ${}^i\mathbf{M}_{c_i}(\tilde{\mathbf{z}}_{i,N+1}, \tilde{\mathbf{z}}_{j,N+1})$, ${}^0\mathbf{F}_{c_j}(\tilde{\mathbf{z}}_{i,N+1}, \tilde{\mathbf{z}}_{j,N+1})$, ${}^j\mathbf{M}_{c_j}(\tilde{\mathbf{z}}_{i,N+1}, \tilde{\mathbf{z}}_{j,N+1})$.

Force/Force-Coupling Approach Using Sequential Integration (Gauss-Seidel Type).**Step 1: Integration of Extrapolated Accelerations**

- Identical to Step 1 in Sect. 9.3.3. (Jacobi-Type)

Step 2: Calculation of Extrapolated Coupling Variables

- Identical to Step 2 in Sect. 9.3.3. (Jacobi-Type), however ${}^0\mathbf{F}_{c_j}^{ex}(t)$ and ${}^j\mathbf{M}_{c_j}^{ex}(t)$ need not to be calculated.

Step 3: Integration of Subsystem 1

- Integrating subsystem 1 from T_N to T_{N+1} with the initial conditions (9.32) and with the extrapolated coupling forces/torques ${}^0\mathbf{F}_{c_i}^{ex}(t)$ and ${}^i\mathbf{M}_{c_i}^{ex}(t)$ of Eq. (9.35), we get the new state variables for subsystem 1 at the macro-time point T_{N+1} , namely

$$\hat{\mathbf{z}}_{1,N+1}.$$

- Update accelerations ${}^0\dot{\mathbf{v}}_{S_i,N+1}$, $\ddot{\mathbf{y}}_{i,N+1}$ of coupling body i with the equations of motion by using the coupling forces/torques ${}^0\mathbf{F}_{c_i}(\tilde{\mathbf{z}}_{i,N+1}, \tilde{\mathbf{z}}_j^{ex}(T_{N+1}))$, ${}^i\mathbf{M}_{c_i}(\tilde{\mathbf{z}}_{i,N+1}, \tilde{\mathbf{z}}_j^{ex}(T_{N+1}))$.

Step 4: Calculation of Interpolated Coupling Variables

- Using the accelerations ${}^0\dot{\mathbf{v}}_{S_i,N+1}$, $\ddot{\mathbf{y}}_{i,N+1}$, ${}^0\dot{\mathbf{v}}_{S_i,N}$, $\ddot{\mathbf{y}}_{i,N}$, ..., ${}^0\dot{\mathbf{v}}_{S_i,N-k+1}$, $\ddot{\mathbf{y}}_{i,N-k+1}$ at the macro-time points T_{N+1} , T_N , ..., T_{N-k+1} , interpolation polynomials ${}^0\dot{\mathbf{v}}_{S_i}^{in}(t)$, $\ddot{\mathbf{y}}_i^{in}(t)$ of degree k can be generated for the time interval $[T_N, T_{N+1}]$.
- An analytical backward integrating of the polynomials ${}^0\dot{\mathbf{v}}_{S_i}^{in}(t)$, $\ddot{\mathbf{y}}_i^{in}(t)$ from T_{N+1} to T_N with the initial conditions ${}^0\mathbf{v}_{S_i,N+1}$, $\dot{\mathbf{y}}_{i,N+1}$, ${}^0\mathbf{r}_{S_i,N+1}$, $\boldsymbol{\gamma}_{i,N+1}$ yields interpolated state variables for the time interval $[T_N, T_{N+1}]$, namely

$${}^0\mathbf{v}_{S_i}^{in}(t), \dot{\mathbf{y}}_i^{in}(t), {}^0\mathbf{r}_{S_i}^{in}(t), \boldsymbol{\gamma}_i^{in}(t). \quad (9.36)$$

- For the simple case of constant approximation ($k = 0$), the interpolated state variables read as

$$\begin{aligned} {}^0\mathbf{v}_{S_i}^{in}(t) &= {}^0\dot{\mathbf{v}}_{S_i,N+1} \cdot (t - T_{N+1}) + {}^0\mathbf{v}_{S_i,N+1}, \\ \dot{\mathbf{y}}_i^{in}(t) &= \ddot{\mathbf{y}}_{i,N+1} \cdot (t - T_{N+1}) + \dot{\mathbf{y}}_{i,N+1} \\ {}^0\mathbf{r}_{S_i}^{in}(t) &= \frac{1}{2} {}^0\dot{\mathbf{v}}_{S_i,N+1} \cdot (t - T_{N+1})^2 + {}^0\mathbf{v}_{S_i,N+1} \cdot (t - T_{N+1}) + {}^0\mathbf{r}_{S_i,N+1}, \\ \boldsymbol{\gamma}_i^{in}(t) &= \frac{1}{2} \ddot{\mathbf{y}}_{i,N+1} \cdot (t - T_{N+1})^2 + \dot{\mathbf{y}}_{i,N+1} \cdot (t - T_{N+1}) + \boldsymbol{\gamma}_{i,N+1}. \end{aligned} \quad (9.37)$$

Step 5: Calculation of Approximated Coupling Forces/Torques

- Inserting the interpolated state variables $\tilde{\mathbf{z}}_i^{in}(t) = \left({}^0\mathbf{r}_{S_i}^{in}(t), \boldsymbol{\gamma}_i^{in}(t), {}^0\mathbf{v}_{S_i}^{in}(t), \dot{\mathbf{y}}_i^{in}(t) \right)^T \in \mathbb{R}^{12}$ of Eq. (9.37) and the corresponding extrapolated state variables $\tilde{\mathbf{z}}_j^{ex}(t)$ of Eq. (9.33) into the constitutive equations (9.29), approximation polynomials of degree $k + 2$ for the time interval $[T_N, T_{N+1}]$ can be calculated for the coupling forces/torques, that is

$${}^0\mathbf{F}_{c_j}^{in}(t) = {}^0\mathbf{F}_{c_j}^{in}(\tilde{\mathbf{z}}_i^{in}(t), \tilde{\mathbf{z}}_j^{ex}(t)), {}^j\mathbf{M}_{c_j}^{in}(t) = {}^j\mathbf{M}_{c_j}^{in}(\tilde{\mathbf{z}}_i^{in}(t), \tilde{\mathbf{z}}_j^{ex}(t)). \quad (9.38)$$

Step 6: Integration of Subsystem 2

- Integrating subsystem 2 from T_N to T_{N+1} with the initial conditions (9.32) and with the approximated coupling forces/torques ${}^0\mathbf{F}_{c_j}^{in}(t)$ and ${}^j\mathbf{M}_{c_j}^{in}(t)$

according to Eq. (9.38) yields the new state variables for subsystem 2 at the macro-time point T_{N+1} , namely

$$\hat{\mathbf{z}}_{2,N+1}.$$

- Finally, an update of the accelerations ${}^0\dot{\mathbf{v}}_{S_i,N+1}$, $\ddot{\mathbf{y}}_{i,N+1}$, ${}^0\dot{\mathbf{v}}_{S_j,N+1}$, $\ddot{\mathbf{y}}_{j,N+1}$ of subsystem 1 and 2 is carried out by means of the equations of motion using the coupling forces/torques ${}^0\mathbf{F}_{c_i}(\tilde{\mathbf{z}}_{i,N+1}, \tilde{\mathbf{z}}_{j,N+1})$, ${}^i\mathbf{M}_{c_i}(\tilde{\mathbf{z}}_{i,N+1}, \tilde{\mathbf{z}}_{j,N+1})$, ${}^0\mathbf{F}_{c_j}(\tilde{\mathbf{z}}_{i,N+1}, \tilde{\mathbf{z}}_{j,N+1})$ and ${}^j\mathbf{M}_{c_j}(\tilde{\mathbf{z}}_{i,N+1}, \tilde{\mathbf{z}}_{j,N+1})$.

9.4 Numerical Examples

In this section, the integrated acceleration co-simulation approach is examined with three nonlinear models.

9.4.1 Nonlinear Two-Mass Oscillator

We consider again the two-mass oscillator depicted in Fig. 9.1. The linear springs/dampers are, however, exchanged by nonlinear springs/dampers with cubic nonlinearities resulting in the following equations of motion

$$\begin{aligned} \dot{x}_1 &= v_1 \\ m_1 \dot{v}_1 &= -c_1 \cdot x_1 - c_{13} \cdot x_1^3 - d_1 \cdot v_1 - d_{13} \cdot v_1^3 + c_c \cdot (x_2 - x_1) \\ &\quad + c_{c3} \cdot (x_2 - x_1)^3 + d_c \cdot (v_2 - v_1) + d_{c3} \cdot (v_2 - v_1)^3 \\ \dot{x}_2 &= v_2 \\ m_2 \dot{v}_2 &= -c_2 \cdot x_2 - c_{23} \cdot x_2^3 - d_2 \cdot v_2 - d_{23} \cdot v_2^3 - c_c \cdot (x_2 - x_1) \\ &\quad - c_{c3} \cdot (x_2 - x_1)^3 - d_c \cdot (v_2 - v_1) - d_{c3} \cdot (v_2 - v_1)^3. \end{aligned} \quad (9.39)$$

In the subsequent investigations, three different cases are considered.

Case 9.1 (Undamped Oscillator) One main advantage of the integrated acceleration co-simulation approach is the significantly improved numerical stability for undamped systems. To demonstrate the applicability of the integrated acceleration approach for conservative systems, the nonlinear two-mass oscillator is considered with the following parameters: $m_1 = 0.1$ kg, $m_2 = 2$ kg, $c_1 = c_2 = 1E3$ N/m, $c_c = 1E4$ N/m, $d_1 = d_2 = d_c = 0$ Ns/m, $c_{13} = c_{23} = 1E3$ N/m³, $c_{c3} = 1E4$ N/m³, $d_{13} = d_{23} = d_{c3} = 0$ Ns³/m³. As initial conditions, $x_{1,0} = x_{2,0} = 0$ m, $v_{1,0} = 100$ m/s and $v_{2,0} = -50$ m/s are chosen. It should be mentioned that the subsystems are integrated numerically with an implicit Runge-Kutta solver. Numer-

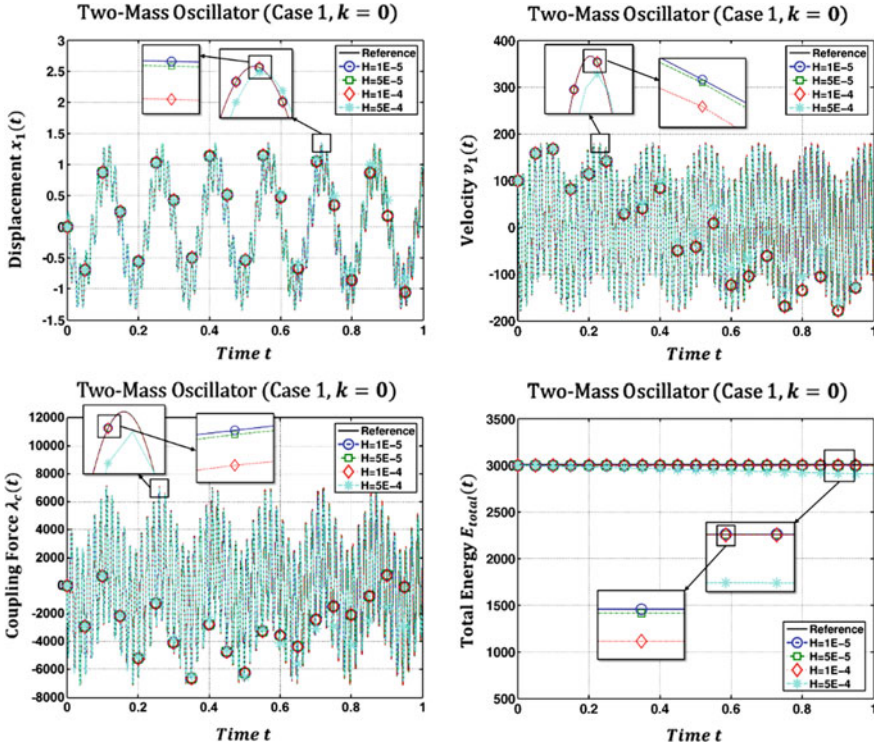


Fig. 9.13 Co-simulation results for the nonlinear two-mass oscillator (force/force-decomposition, Jacobi type, $k = 0$, different macro-step sizes H): displacement $x_1(t)$, velocity $v_1(t)$, coupling force $\lambda_c(t)$ and total energy $E_{total}(t)$

ical results generated with the integrated acceleration co-simulation approach based on force/force-decomposition (Jacobi type, $k = 0$) are collected in Fig. 9.13. The plot depicts the displacement $x_1(t)$ and the velocity $v_1(t)$ of mass 1, the coupling force $\lambda_c(t) = c_c \cdot (x_2 - x_1) + c_{c3} \cdot (x_2 - x_1)^3$ and the total energy $E_{total}(t)$ of the system (sum of kinetic and potential energy) for different macro-step sizes H . As can be seen, the force/force-coupling approach yields stable results for the undamped oscillator. Numerical dissipation is small and can be reduced by decreasing the macro-step size. Corresponding simulations for the case $k = 1$ are collected in Fig. 9.14. Convergence plots for the undamped nonlinear two-mass oscillator are depicted in Fig. 9.15.

Case 9.2 (Slightly Damped Oscillator) Next, the nonlinear oscillator is considered with small damping. Simulations are carried out with the parameters $m_1 = 0.1$ kg, $m_2 = 2$ kg, $c_1 = c_2 = 1E3 \frac{N}{m}$, $c_c = 1E4 \frac{N}{m}$, $d_1 = d_2 = d_c = 0.1 \frac{Ns}{m}$, $c_{13} = c_{23} = c_{c3} = 1E3 \frac{N}{m^3}$, $d_{13} = d_{23} = d_{c3} = 1E - 5 \frac{Ns^3}{m^3}$ and the initial conditions $x_{1,0} = 0$ m, $x_{2,0} = 1$ m, $v_{1,0} = v_{2,0} = 0$ m/s. Numerical results calculated with the integrated acceleration co-simulation approach are arranged in Fig. 9.16 for all three

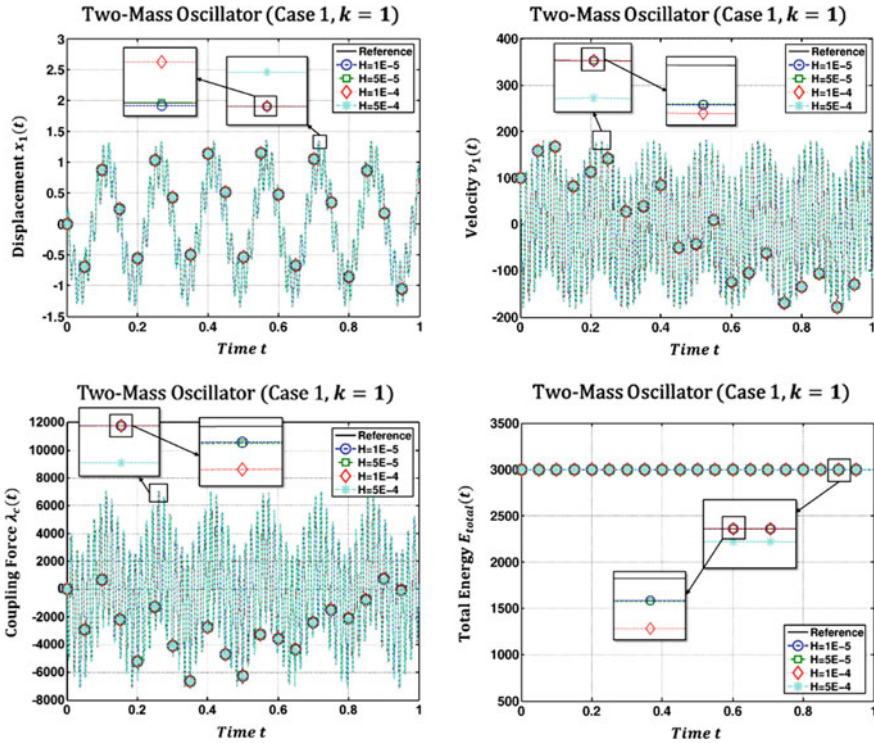


Fig. 9.14 Co-simulation results for the nonlinear two-mass oscillator (force/force-decomposition, Jacobi type, $k = 1$, different macro-step sizes H): displacement $x_1(t)$, velocity $v_1(t)$, coupling force $\lambda_c(t)$ and total energy $E_{total}(t)$

decomposition techniques (Jacobi and Gauss-Seidel type, $k = 0$, macro-step size $H = 1E - 4s$). The plots depict the displacement $x_1(t)$, the velocity $v_1(t)$ and the coupling force $\lambda_c(t)$. As can be seen, all coupling methods yield stable results. The results of the co-simulation are close to the reference solution. The corresponding convergence plots for the slightly damped oscillator are arranged in Fig. 9.17.

Case 9.3 (Highly Damped Oscillator) Explicit coupling schemes usually fail or require rather small macro-step sizes for very stiff systems (characterized by large values of $\bar{\Lambda}_{i1}$) and for highly-damped systems (characterized by large values of $\bar{\Lambda}_{r1}$). The stability plots of Sect. 9.2.4 (displacement/displacement-coupling), however, indicate that highly-damped systems may be integrated stable and with moderate macro-sizes if a sequential displacement/displacement-decomposition approach is applied. To show the improved stability of this coupling technique, we regard the nonlinear two-mass oscillator with the subsequent parameters: $m_1 = 0.1$ kg, $m_2 = 2$ kg, $c_1 = c_{13} = c_2 = c_{23} = c_c = c_{c3} = 1E3$, $d_1 = d_{13} = d_c = d_{c3} = 1E3$, $d_2 = d_{23} = 1E6$. As initial conditions, $x_{1,0} = 1$ m, $x_{2,0} = -0.5$ m, $v_{1,0} = v_{2,0} = 0$ m/s are chosen. In Fig. 9.18, $x_1(t)$, $v_1(t)$ and $\lambda_c(t)$ are shown, which

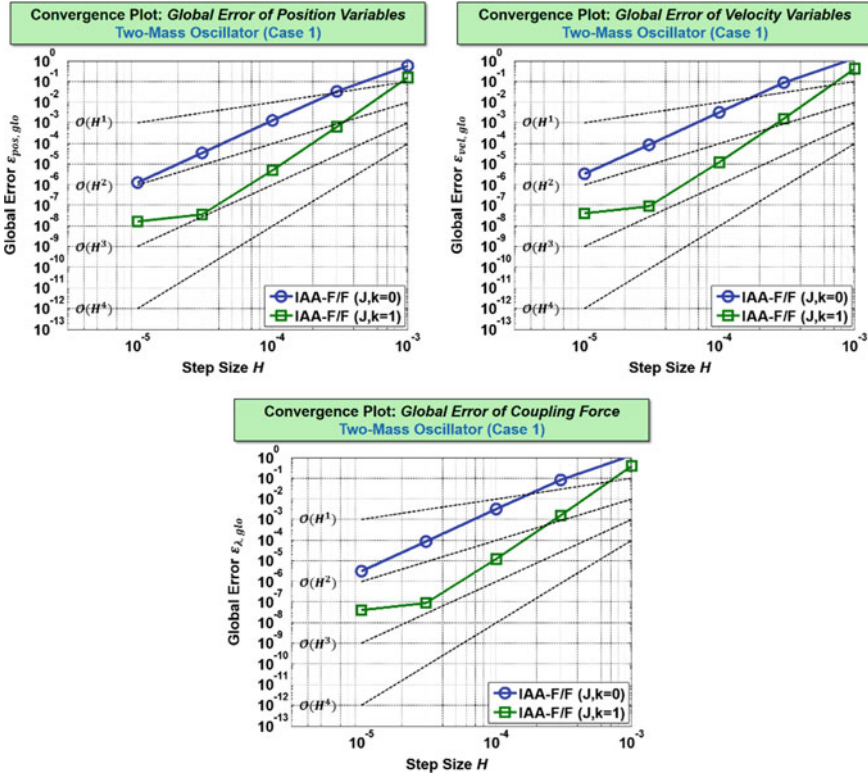


Fig. 9.15 Convergence plots for the undamped nonlinear two-mass oscillator (force/force-decomposition, Jacobi type, $k = \mathbf{0}$ and $k = \mathbf{1}$): global error of the position variables, the velocity variables and the coupling force over the macro-step size H

were calculated with the integrated acceleration co-simulation approach based on a displacement/displacement-decomposition approach (Gauss-Seidel type, $k = 0$, $H = 1E - 3s$). As can be seen, the co-simulation method is numerically stable. Corresponding simulations with the parallel displacement/displacement-decomposition approach and with the force/force—and force/displacement-coupling approaches (Jacobi and Gauss-Seidel type) are unstable for $H = 1E - 3s$. In Fig. 9.19, convergence plots for the highly damped oscillator are shown.

9.4.2 Spherical 4-Bar Mechanism

In a second example, the spherical 4-bar mechanism depicted in Fig. 9.20 is analyzed. The system can be regarded as two spherical double pendulums, which are connected by a compliant joint. The flexible joint is modeled by a 3D linear spring/damper

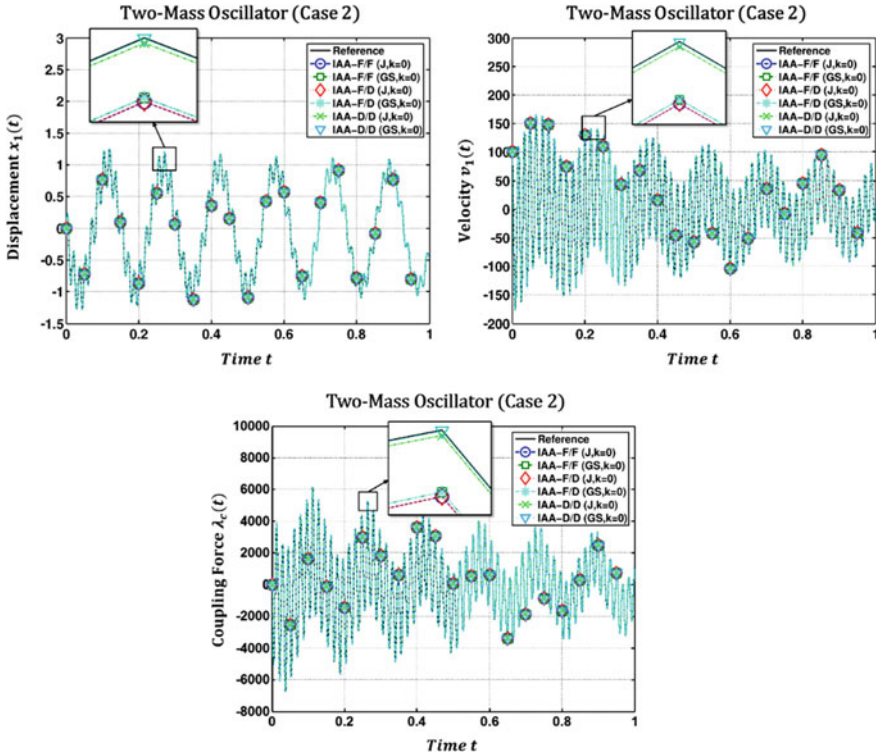


Fig. 9.16 Co-simulation results for the nonlinear two-mass oscillator (all three decomposition approaches, Jacobi and Gauss-Seidel type, $k = 0$, macro-step size $H = 1E - 4$): displacement $x_1(t)$, velocity $v_1(t)$ and coupling force $\lambda_c(t)$

system. The bars are assumed to be thin links (circular cross section, radius r) with the moments of inertia $J_1 = J_2 = 1/12 m l^2$ and $J_3 = 1/2 m r^2$ with respect to the body fixed principal axes system at the center of mass. Bar 1 is connected to ground by a revolute joint; bar 2 and bar 1 are coupled by a spherical joint. Bar 3 and bar 4 are connected by an universal joint; bar 4 is attached to ground by means of a revolute joint. Subsystem 1 consists of bar 1 and bar 2; subsystem 2 is constituted by bar 3 and bar 4. The joints are assumed to be ideal rigid joints so that each subsystem is described by a nonlinear system of differential-algebraic equations. The two subsystems are coupled by a linear 6-DOF bushing element, represented by the three translational stiffnesses $c_{cx} = c_{cy} = c_{cz} = 1E3$ N/m and the corresponding damping coefficients $d_{cx} = d_{cy} = d_{cz} = 5$ Ns/m. The rotational stiffnesses are assumed to be $c_{c\alpha_x} = c_{c\alpha_y} = c_{c\alpha_z} = 1E2$ Nm/rad and the related damping coefficients $d_{c\alpha_x} = d_{c\alpha_y} = d_{c\alpha_z} = 0$ Nms/rad. A linear viscous damping force is applied at the center of mass of all four bars (damping coefficients $d_x = d_y = d_z = 0.1$ Ns/m). The system is driven by the externally applied torque ${}^1M_{z_1}(t) = 1 \cdot t$ Nm acting at the center of mass of body 1 in body-fixed z_1 -direction. For the numerical calculations,

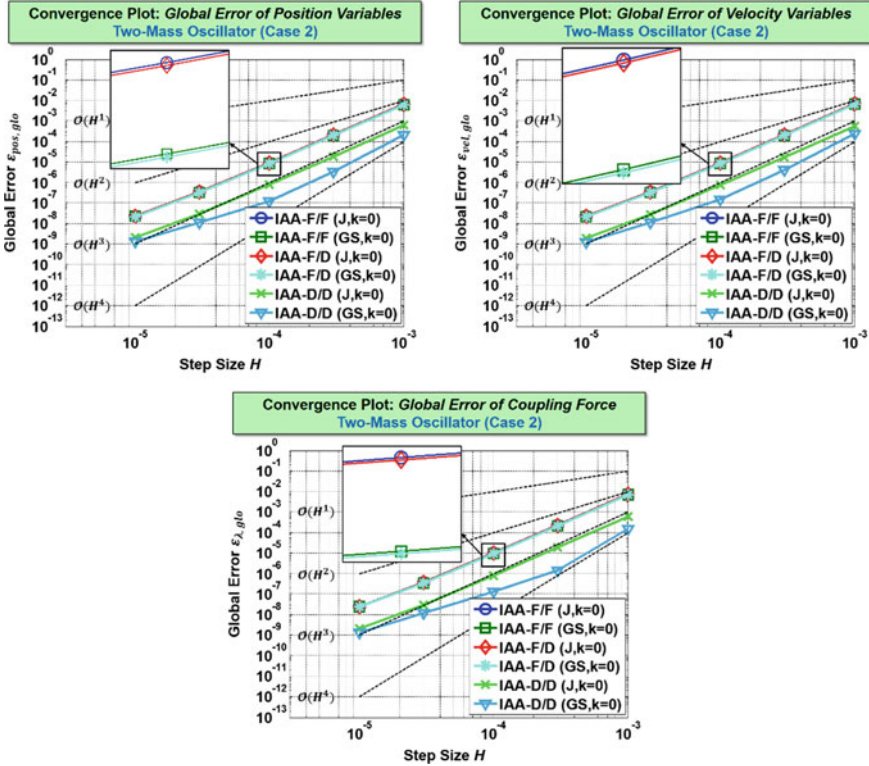


Fig. 9.17 Convergence plots for the slightly damped nonlinear two-mass oscillator (all three decomposition approaches, Jacobi and Gauss-Seidel type, $k = 0$): global error of the position variables, the velocity variables and the coupling force over the macro-step size H

the parameters $m_1 = 1/2$ kg, $m_2 = \sqrt{2}$ kg, $m_3 = 3/2$ kg, $l_1 = 1/2$ m, $l_2 = \sqrt{2}$ m, $l_3 = 3/2$ m, $r = 0.1$ m were used. The co-simulations were carried out with the integrated acceleration approaches described in Sects. 9.2 and 9.3. The macro-step size is assumed to be constant, namely $H = 1E - 3s$. Results are presented for all three decomposition techniques (Jacobi and Gauss-Seidel type) using constant ($k = 0$) and linear ($k = 1$) approximation polynomials. At the beginning of the simulation, the system is assumed to be at rest in the sketched upright position.

Simulation results for constant approximation polynomials ($k = 0$) are collected in Fig. 9.21, which shows the displacement $x_{S_2}(t)$ in x -direction and the corresponding velocity $v_{S_2}(t) = \dot{x}_{S_2}(t)$ of the center of mass S_2 of bar 2. Moreover, the first Bryant angle $\alpha_2(t)$ and the angular velocity ${}^2\omega_{x_2}(t)$ of bar 2 are plotted. The figure also depicts the coupling force ${}^0F_{cx}(t)$ acting in space-fixed x -direction and the coupling torque ${}^2M_{cx_2}(t)$ acting in the body-fixed x_2 -direction. Corresponding results for the case of linear approximation polynomials ($k = 1$) are collected in Fig. 9.22. Convergence plots are shown in Fig. 9.23. Within the considered time range of 5 s, all

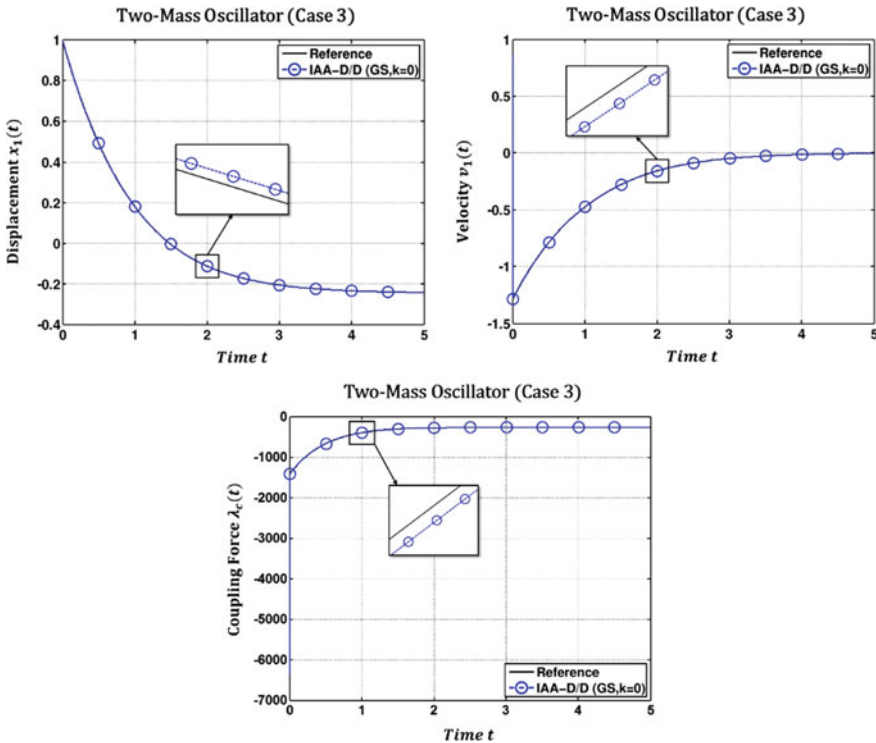


Fig. 9.18 Co-simulation results for the nonlinear two-mass oscillator (displacement/displacement-decomposition approach, Gauss-Seidel type, $k = 0$, macro-step size $H = 1E - 3$): displacement $x_1(t)$, velocity $v_1(t)$ and coupling force $\lambda_c(t)$

coupling approaches yield accurate results. Numerical instabilities, i.e. blowing-up phenomena, are not observed. Simulations with a longer simulation time, which are not shown here for the reason of a concise representation, also exhibit only stable results for the considered system, i.e. no system blow-up until the final simulation time. It should, however, be pointed out that an analytical proof for stability is not possible for this example, since the model is highly non-linear. Strictly speaking, the stability analysis of Sect. 9.2 can only be applied for linear problems. Linearizing, however, the non-linear model about an equilibrium point, for instance, and using the stability plots of Sect. 9.2 may give an “impression” on the macro-step size required for getting stable results. As a matter of fact, one is faced with the same problem in connection with time integrators, since Dahlquist’s stability theory is also only valid for the linear case.

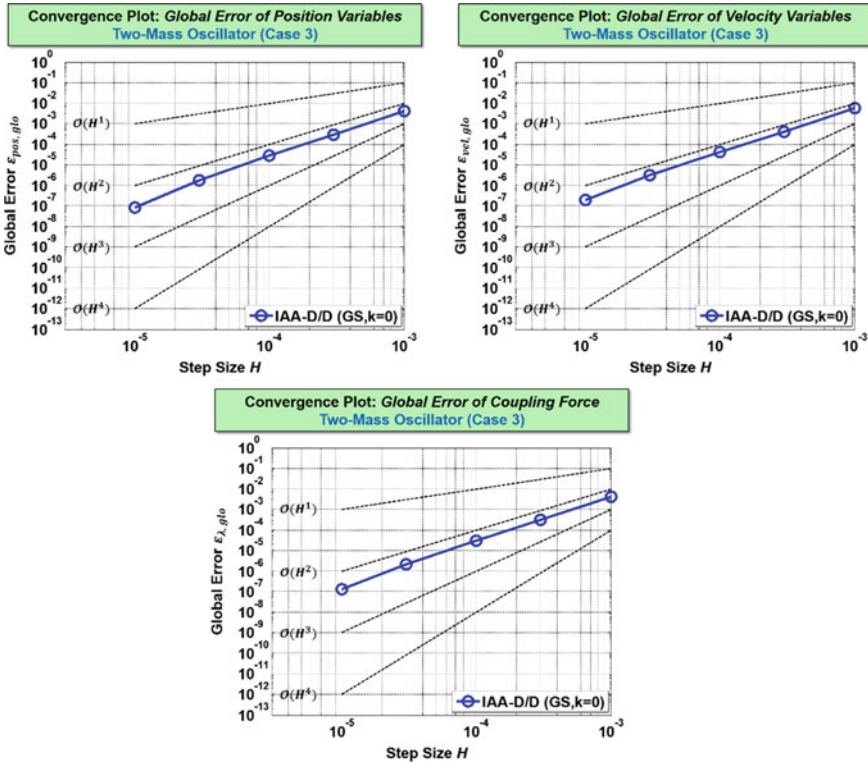


Fig. 9.19 Convergence plots for the slightly damped nonlinear two-mass oscillator (displacement/displacement-decomposition approach, Gauss-Seidel type, $k = 0$): global error of the position variables, the velocity variables and the coupling force over the macro-step size H

9.4.3 Front Wheel Suspension: Coupled MBS/FEM-Simulation

In a final example, a coupled multibody/finite-element system is investigated. We consider the double-wishbone front wheel suspension depicted in Fig. 9.24. The suspension—except for the lower arm and the suspension strut—is modeled as a 3D multibody system. The lower arm is represented by a 3D nonlinear finite element body (discretization with approx. 10,000 hexahedral elements). At the coupling point C , the lower arm is connected to the multibody system by a bushing element (coupling stiffnesses $c_{cx} = c_{cy} = c_{cz} = 1E5N/m$, coupling damping coefficients $d_{cx} = d_{cy} = d_{cz} = 1E2Ns/m$). The wheel is base-point excited by the step-shaped displacement $u_{ex}(t) = \frac{0.05}{\pi} \cdot \{ \text{atan}[100 \cdot (t - 0.5)] - \text{atan}[100 \cdot (t - 1.5)] \} m$ illustrated in Fig. 9.24, which models the driving over an obstacle. The multibody system (subsystem 1) is integrated with an index-2 BDF-integrator (error tolerance $1E-6$). The geometrically nonlinear finite element system (subsystem 2) is integrated with the explicit central

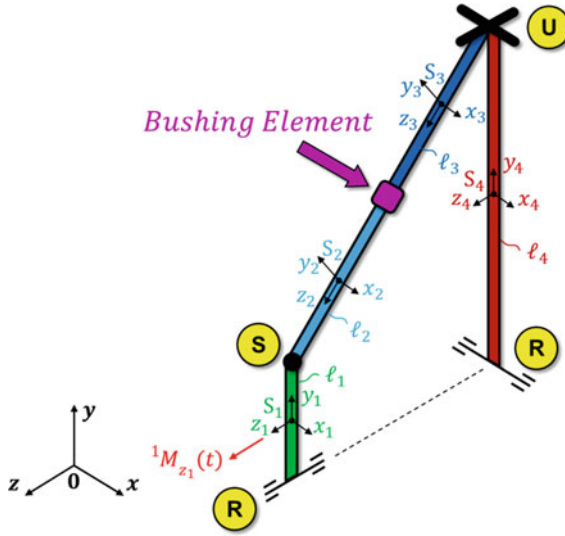


Fig. 9.20 Spherical 4-bar mechanism: interpretation as two spherical double pendulums coupled by a bushing element

difference method (fixed integration step-size $h \approx 1.9E - 7s$). The co-simulation is carried out with the integrated acceleration approach (force/force-coupling, Jacobi-type, $k = 0$) using the constant macro-step sizes $H = 5E - 6s$, $H = 1E - 6s$, $H = 5E - 7s$ and $H = 1E - 7s$.

Simulation results are collected in Fig. 9.25. The figure shows the displacements $u_{Cx}(t)$, $u_{Cy}(t)$, $u_{Cz}(t)$ of the coupling point C in global x -, y -, z -direction as well as the coupling forces ${}^0F_{cx}(t)$, ${}^0F_{cy}(t)$ and ${}^0F_{cz}(t)$. As can be seen, the coupling approach yields stable results; the curves for the four macro-step sizes are very close together, indicating the good convergence of the coupling approach.

9.5 Conclusions

A novel explicit co-simulation approach with an improved stability and convergence behavior has been presented in this manuscript. In the framework of this manuscript, we have assumed that the coupling between the subsystems is described by constitutive equations so that the subsystems are coupled by applied forces/torques. In classical co-simulation approaches, the coupling variables are directly approximated using Lagrange polynomials, for instance. Here, the acceleration variables (acceleration coordinates of the center of mass; second derivative of the rotation parameters) of the coupling bodies are approximated with Lagrange polynomials. To get extrapolation/interpolation polynomials for the coupling variables (e.g., cou-

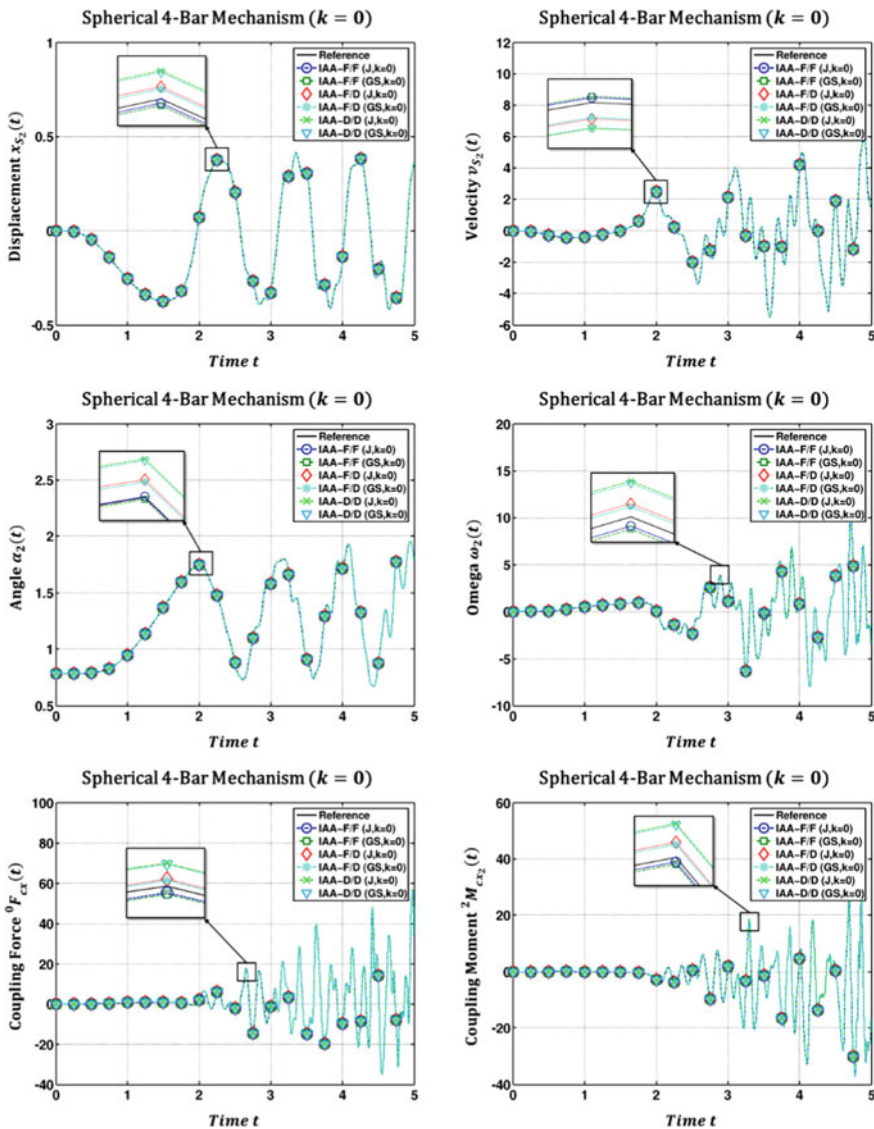


Fig. 9.21 Simulation results for the 4-bar mechanism (all three decomposition approaches, Jacobi and Gauss-Seidel type, $k = 0$, macro-step size $H = 1E - 4$): displacement $x_{S_2}(t)$ and velocity $v_{S_2}(t)$, Bryant angle $\alpha_2(t)$ and angular velocity ${}^2\omega_{2x}(t)$, coupling force ${}^0F_{cx}(t)$ and coupling torque ${}^2M_{cx_2}(t)$

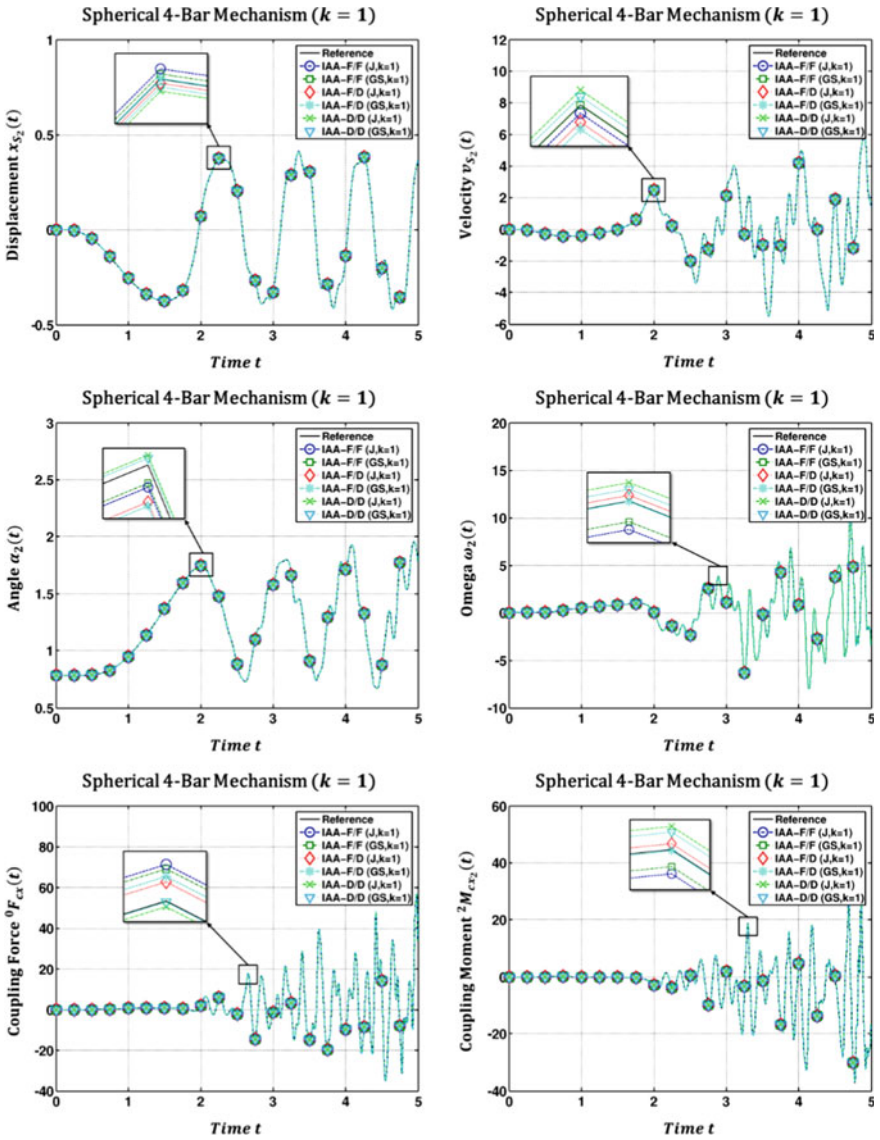


Fig. 9.22 Simulation results for the 4-bar mechanism (all three decomposition approaches, Jacobi and Gauss-Seidel type, $k = 1$, macro-step size $H = 1E - 4$): displacement $x_{S_2}(t)$ and velocity $v_{S_2}(t)$, Bryant angle $\alpha_2(t)$ and angular velocity ${}^2\omega_{2x}(t)$, coupling force ${}^0F_{cx}(t)$ and coupling torque ${}^2M_{cx_2}(t)$

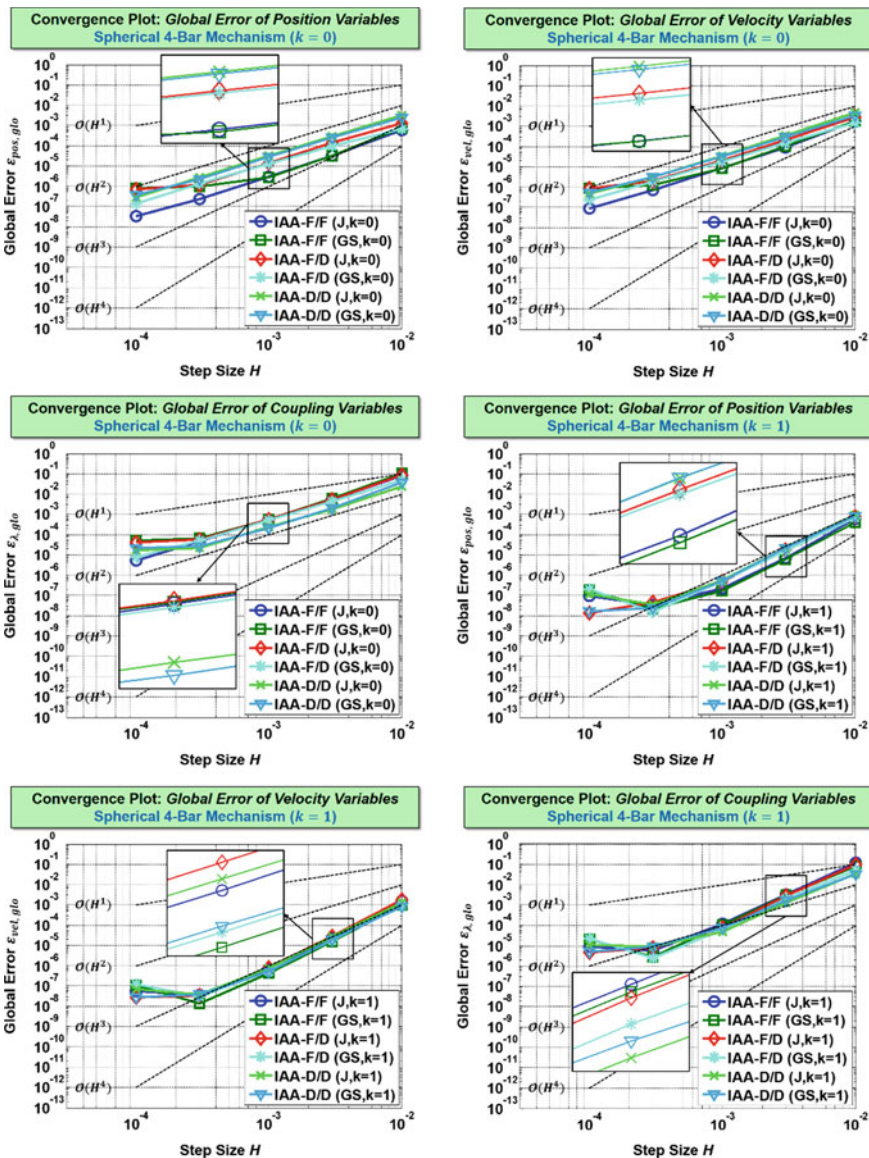


Fig. 9.23 Convergence plots for the spherical 4-bar mechanism (all three decomposition approaches, Jacobi and Gauss-Seidel type, $k = 0$ and $k = 1$): global error of the position variables, the velocity variables and the coupling force over the macro-step size H

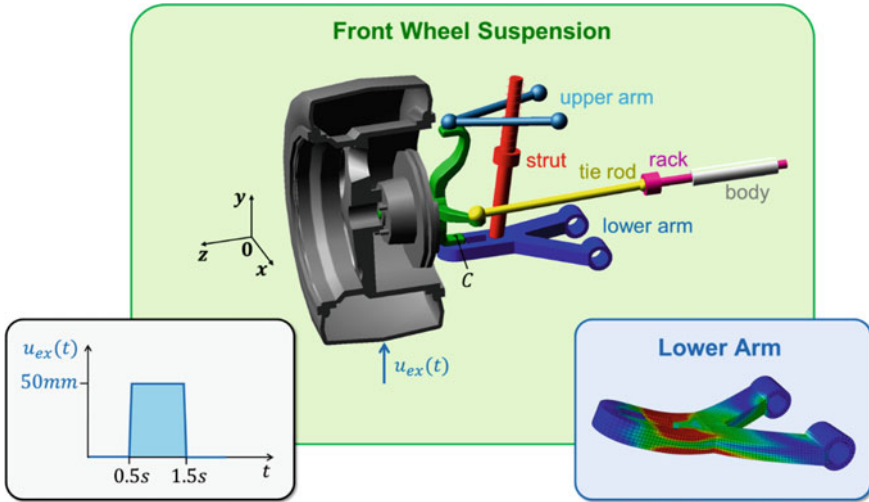


Fig. 9.24 Front wheel suspension: coupled multibody/finite-element system

pling forces/torques in case of a force/force-coupling approach), the approximated accelerations are analytically integrated in order to get approximation polynomials for the state variables of the two coupling bodies. Inserting the approximated state variables into the constitutive equations yields approximation polynomials for the coupling variables.

Good results with respect to the numerical stability and convergence behavior are already achieved, if constant polynomials ($k = 0$) are used for approximating the accelerations. For $k = 0$, the global errors of the position and velocity variables converge with $\mathcal{O}(H^2)$ and the related local errors with $\mathcal{O}(H^4)$ and $\mathcal{O}(H^3)$, respectively. Using a classical explicit co-simulation approach with linear approximation polynomials, the same convergence orders are achieved, however with significantly larger error magnitudes. The main advantage of the presented approach is, however, the increased numerical stability compared to the classical explicit approach. One drawback of classical coupling techniques is the small—or even non existing—region of stability for undamped (slightly damped) systems. For systems without damping, the new approach in connection with force/force-decomposition may, however, be applied very advantageously, since the region of stability is significantly increased especially for undamped systems. A further advantage of the integrated acceleration approach for $k = 0$ compared to the classical approach for $k = 1$ is the fact that the integrated acceleration algorithm is self-starting, i.e. there is no initialization process necessary at the beginning of the co-simulation.

Comparing the integrated acceleration approach with linear approximation polynomials ($k = 1$) for the acceleration variables and its classical counterpart with quadratic approximation polynomials ($k = 2$) for the coupling variables, the improved performance of the new approach is even more significant (see Appendix 1).

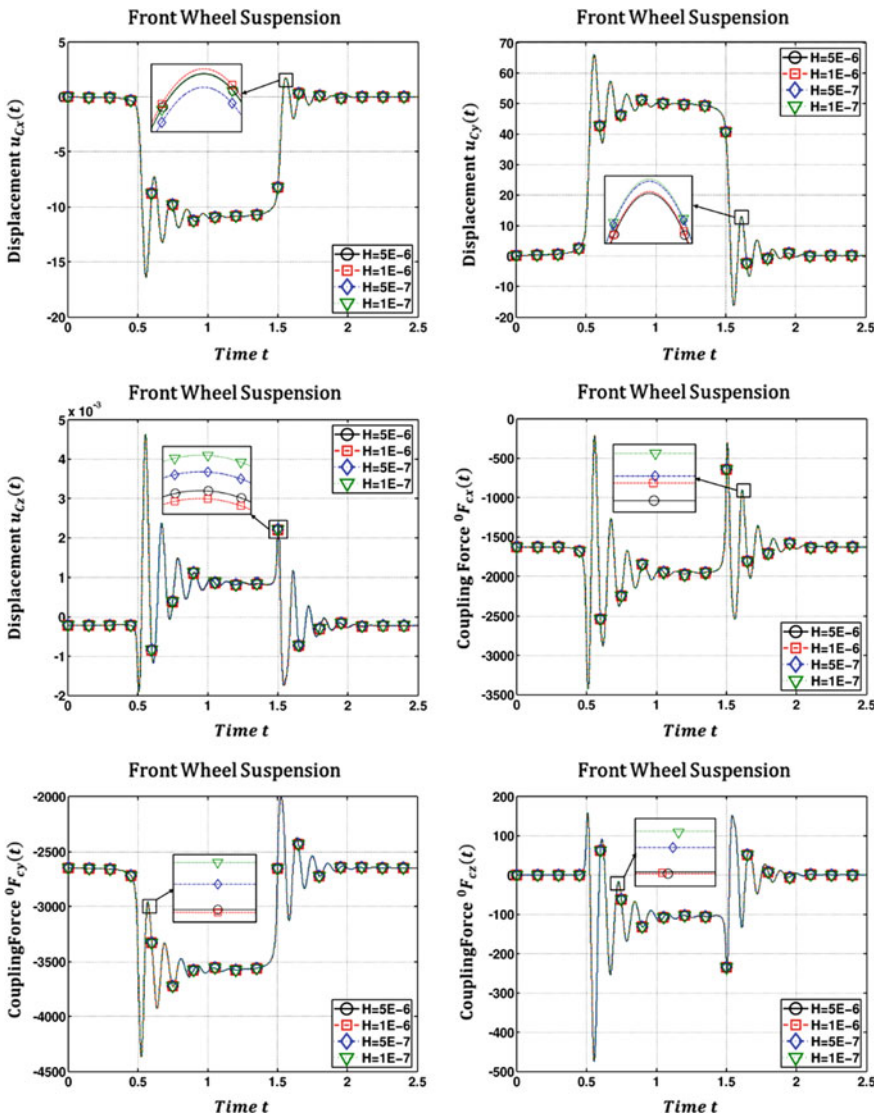


Fig. 9.25 Co-simulation results for the front wheel suspension: displacements $u_{Cx}(t), u_{Cy}(t), u_{Cz}(t)$ of coupling point C and coupling forces ${}^0F_{Cx}(t), {}^0F_{Cy}(t)$ and ${}^0F_{Cz}(t)$

While the integrated acceleration approaches for $k = 0$ and $k = 1$ exhibit a similar stability behavior (at least for the symmetrical test model with $\alpha_{m21} = \alpha_{\Lambda r21} = \alpha_{\Lambda i21} = \alpha_{\Lambda rc1} = \alpha_{\Lambda ic1} = 1$), the convergence order and in consequence the accuracy of the co-simulation method is markedly improved if the accelerations are approximated linearly.

It should finally also be mentioned that the new approaches have some drawbacks compared with the classical approaches. Firstly, an update step for calculating the accelerations is necessary at each macro-time point, which entails an additional implementation effort. Secondly, the integrated acceleration approach can only be applied, if constitutive laws describing the coupling between the subsystems are known. This is not required for the classical approaches.

Appendix 1: Stability and Convergence Plots for Linear Approximation Polynomials

In this appendix, stability and convergence plots are collected for the case that linear approximation polynomials ($k = 1$) are used for approximating the acceleration variables. The stability plots for the case $k = 1$, see Fig. 9.26, resemble the corresponding plots for $k = 0$ in Sect. 9.2.4. Compared with its classical counterpart based on quadratic approximation polynomials for the coupling variables, see Fig. 8 in Ref. [71], the integrated acceleration approach for $k = 1$ shows a significantly better numerical stability behavior.

A convergence analysis exhibits the improved convergence behavior for the case that linear approximation polynomials are used, see Fig. 9.27. For $k = 1$, the global errors $\varepsilon_{pos, glo}$ and $\varepsilon_{vel, glo}$ converge with $\mathcal{O}(H^3)$; the local errors $\varepsilon_{pos, loc}$ converge with $\mathcal{O}(H^5)$ and $\varepsilon_{vel, loc}$ with $\mathcal{O}(H^4)$. Note that the same convergence behavior—however with larger error magnitudes—is achieved with the classical approaches using quadratic approximation polynomials for the coupling variables, see Fig. 12 in Ref. [71].

Appendix 2: Stability Plots for Displacement/Force-Coupling (Gauss-Seidel Type)

While force/force- and displacement/displacement-coupling represent symmetrical decomposition techniques, force/displacement-coupling is an unsymmetrical decomposition approach. Considering the co-simulation test model, one has therefore to distinguish, whether subsystem 1 or subsystem 2 is the force-excited single-mass oscillator (“force/displacement-coupling (F/D)” or “displacement/force-coupling (D/F)”), see Figs. 9.3 and 9.28. Using a parallel integration scheme (Jacobi type), the D/F co-simulation test model can be represented by the F/D co-simulation test model simply

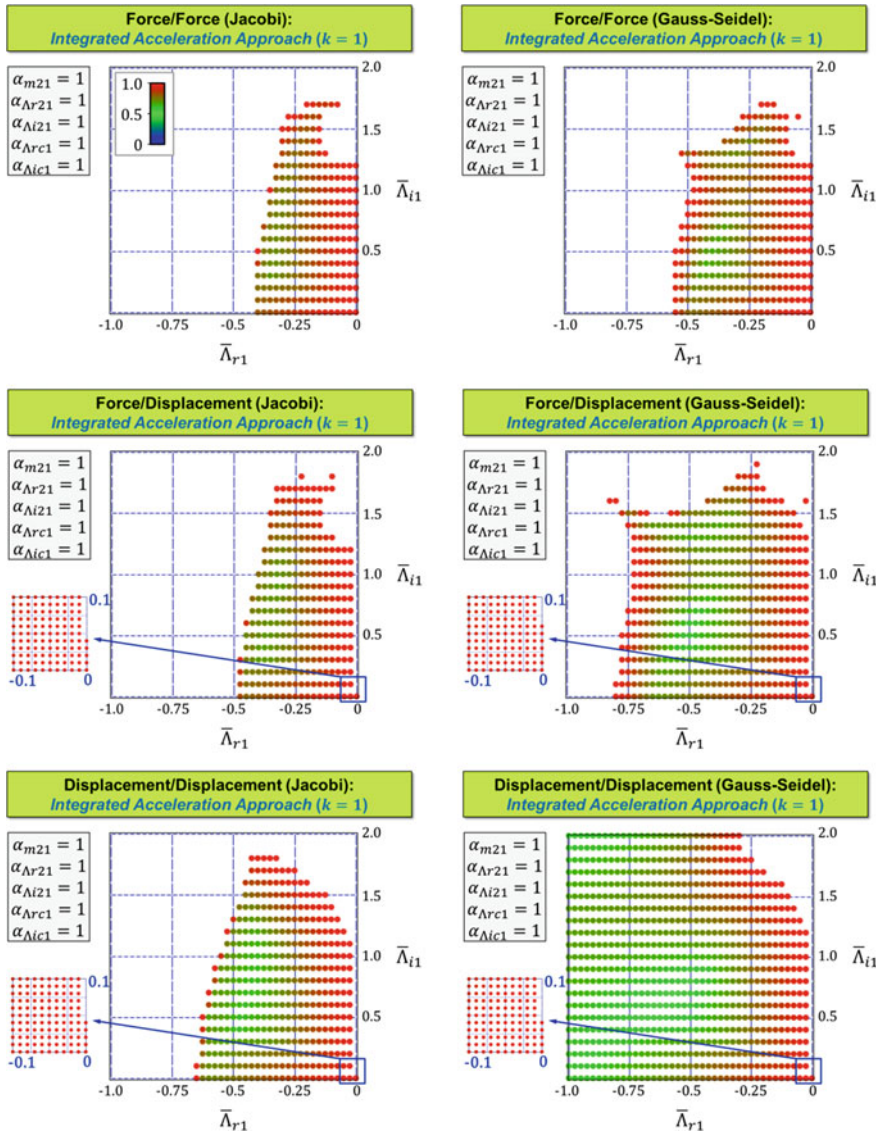


Fig. 9.26 Stability plots for the integrated acceleration co-simulation approach based on linear approximation polynomials ($k = 1$): Jacobi and Gauss-Seidel type using force/force-, force/displacement- and displacement/displacement-decomposition

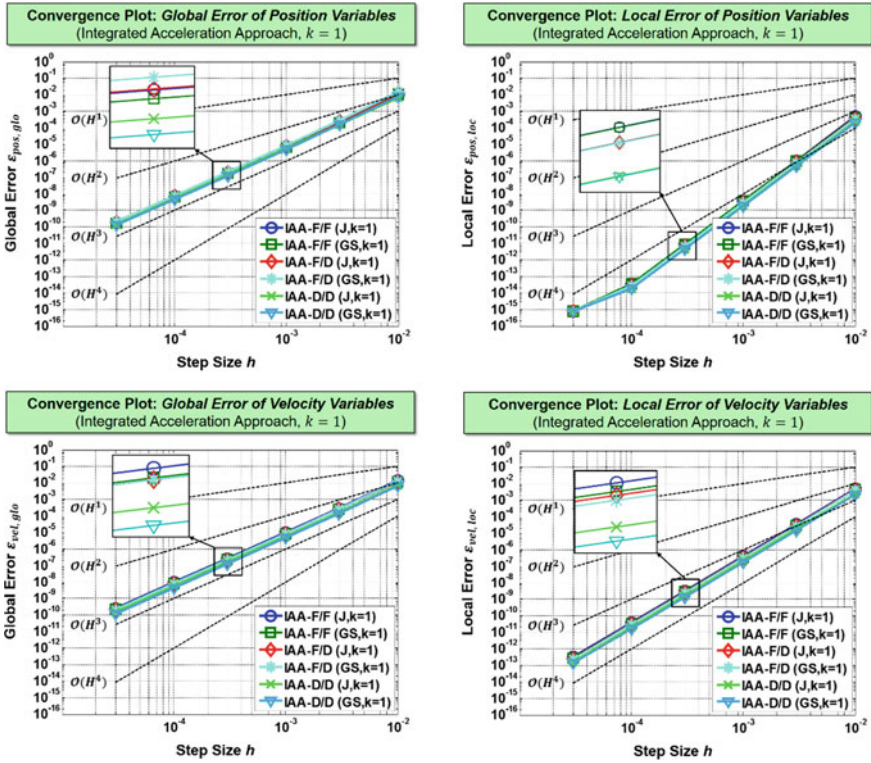


Fig. 9.27 Convergence plots for the integrated acceleration co-simulation approach ($k = 1$): global and local error of the position and velocity variables over the macro-step size H

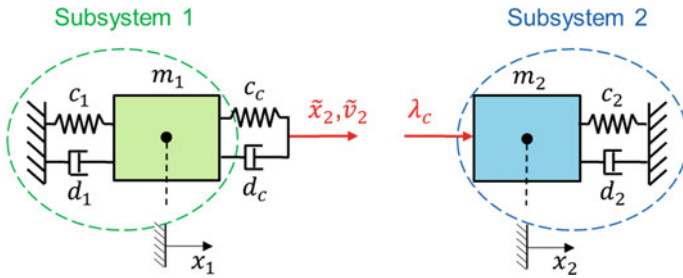


Fig. 9.28 Co-simulation test model: displacement/force-decomposition approach

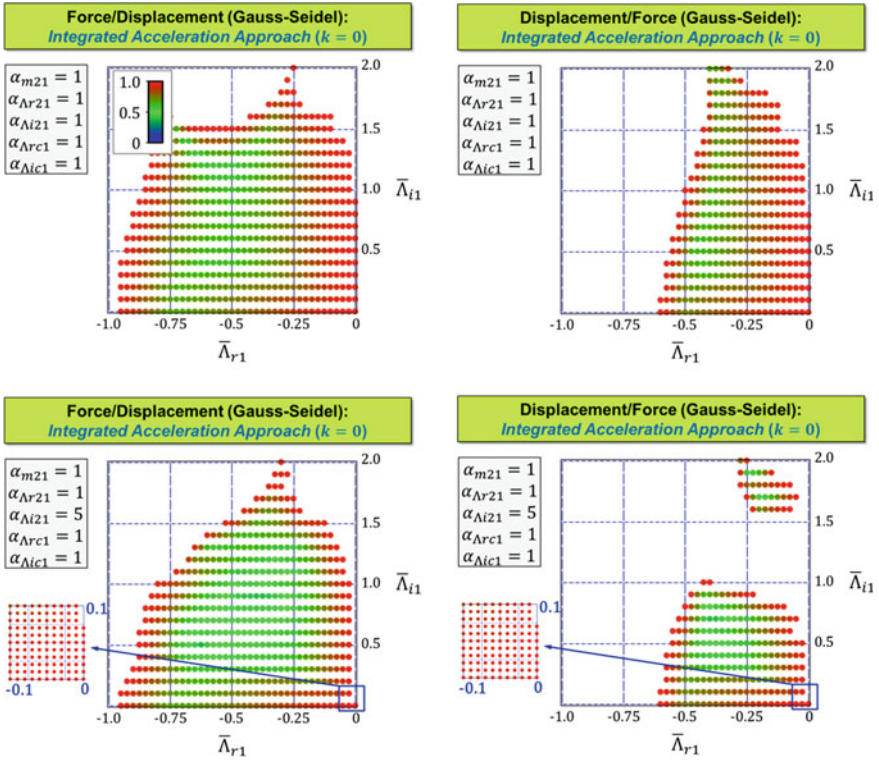


Fig. 9.29 Stability plots for the integrated acceleration co-simulation approach based on force/displacement- and displacement/force-decomposition (Gauss-Seidel type, $k = 0$) for two different test model parameters

by changing the parameters m_1, c_1, d_1 and m_2, c_2, d_2 . For instance, F/D-coupling with $m_1 = 2 \cdot m_2 = m, c_1 = 2 \cdot c_2 = c, d_1 = 2 \cdot d_2 = d, c_c, d_c$ is equivalent with D/F-coupling using $2 \cdot m_1 = m_2 = m, 2 \cdot c_1 = c_2 = c, 2 \cdot d_1 = d_2 = d, c_c, d_c$ provided that the subsystems are integrated in parallel.

For the sequential Gauss-Seidel scheme, however, the D/F test model cannot be represented by the F/D test model by simply changing the model parameters. Using the same test model parameters, D/F-coupling and F/D-coupling may show a different numerical stability behavior. Figure 9.29 shows stability plots for the integrated acceleration approach using F/D- and D/F-decomposition (Gauss-Seidel type, $k = 0$). Two different parameter sets for the test model are investigated. It is interesting to notice that the F/D-coupling approach shows an improved stability behavior for both parameter sets, especially for the case that subsystem 2 is much stiffer than subsystem 1 ($\alpha_{Li21} = 5$).

References

1. Alioli, M., Morandini, M., Masarati, P.: Coupled multibody-fluid dynamics simulation of flapping wings. In: Proceedings of the ASME IDETC/CIE 2013, 4–7 August, Portland, Oregon, USA, DETC2013-12198 (2013)
2. Ambrosio, J., Pombo, J., Rauter, F., Pereira, M.: A memory based communication in the co-simulation of multibody and finite element codes for pantograph-catenary interaction simulation. In: Bottasso, C.L. (ed.) *Multibody Dynamics: Computational Methods and Applications*, pp. 231–252, Springer, Berlin (2009)
3. Ambrosio, J., Rauter, F., Pombo, J., Pereira, M.: Co-simulation procedure for the finite element and flexible multibody dynamic analysis. In: Proceedings of PACAM XI, 11th Pan-American Congress of Applied Mechanics, 04–08 Jan, Foz do Iguacu, PR, Brazil (2010)
4. Ambrosio, J., Pombo, J., Pereira, M., Antunes, P., Mosca, A.: A computational procedure for the dynamic analysis of the catenary-pantograph interaction in high-speed trains. *J. Theor. Appl. Mech.* **50**(3), 681–699 (2012) (Warsaw)
5. Anderson, K.S.: An order- n formulation for the motion simulation of general multi-rigid-body tree systems. *Comput. Struct.* **46**(3), 547–559 (1993)
6. Anderson, K.S., Duan, S.: A hybrid parallelizable low-order algorithm for dynamics of multi-rigid-body systems: part I, chain systems. *Math. Comput. Modell.* **30**(9–10), 193–215 (1999)
7. Arnold, M.: Multi-rate time integration for large scale multibody system models. In: Eberhard, P. (ed.) *Multiscale Problems in Multibody System Contacts*, pp. 1–10. Springer, New York (2007)
8. Arnold, M.: Stability of sequential modular time integration methods for coupled multibody system models. *J. Comput. Nonlinear Dyn.* **5**, 1–9 (2010)
9. Arnold, M., Clauss, C., Schierz, T.: Error analysis and error estimates for co-simulation in FMI for model exchange and co-simulation in V2.0. *Arch. Mech. Eng.* **60**(1), 75–94 (2013)
10. Belytschko, T., Lu, Y.Y.: An explicit multi-time step integration for parabolic and hyperbolic systems. *New Methods Trans. Anal. PVP-Vol. 246/AMD-Vol. 143*, 25–39 (ASME, New York) (1992)
11. Betsch, P.: The discrete null space method for the energy consistent integration of constrained mechanical systems. Part I: holonomic constraints. *Comput. Methods Appl. Mech. Eng.* **194**(50–52), 5159–5190 (2005)
12. Betsch, P., Leyendecker, S.: The discrete null space method for the energy consistent integration of constrained mechanical systems. Part II: multibody dynamics. *Int. J. Numer. Meth. Eng.* **67**, 499–552 (2006)
13. Betsch, P., Steinmann, P.: A DAE approach to flexible multibody dynamics. *Multibody Syst. Dyn.* **8**, 367–391 (2002)
14. Betsch, P., Hesch, C., Sanger, N., Uhlar, S.: Variational integrators and energy-momentum schemes for flexible multibody dynamics. *J. Comput. Nonlinear Dyn.* **5**(3), 031001/1–031001/11 (2010)
15. Betsch, P., Siebert, R.: Rigid body dynamics in terms of quaternions: Hamiltonian formulation and conserving numerical integration. *Int. J. Numer. Meth. Eng.* **79**(4), 444–473 (2009)
16. Betsch, P., Uhlar, S.: Energy-momentum conserving integration of multibody dynamics. *Multibody Syst. Dyn.* **17**(4), 243–289 (2007)
17. Bruls, O., Cardona, A., Arnold, M.: Lie group generalized- α time integration of constrained flexible multibody systems. *Mech. Mach. Theory* **48**, 121–137 (2012)
18. Bruls, O., Golinva, J.C.: The generalized- α method in mechatronic applications. *Z. Angew. Math. Mech.* **86**, 748–758 (2006)
19. Bruls, O., Golinva, J.C.: On the numerical damping of time integrators for coupled mechatronic systems. *Comput. Methods Appl. Mech. Eng.* **32**, 212–227 (2006)
20. Bruls, O., Arnold, M.: The generalized- α scheme as a linear multi-step integrator: towards a general mechatronic simulator. *J. Comput. Nonlinear Dyn.* **3**, 41–57 (2008)
21. Busch, M., Schweizer, B.: Coupled simulation of multibody and finite element systems: an efficient and robust semi-implicit coupling approach. *Arch. Appl. Mech.* **82**(6), 723–741 (2012)

22. Busch, M., Schweizer, B.: An explicit approach for controlling the macro-step size of co-simulation methods. In: Proceedings of The 7th European Nonlinear Dynamics, ENOC 2011, Rome, Italy, 24–29 July (2011)
23. Cardona, A., Geradin, M.: Time integration of the equations of motion in mechanism analysis. *Comput. Struct.* **33**, 801–820 (1998)
24. Chung, J., Hulbert, G.M.: A time integration algorithm for structural dynamics with improved numerical dissipation: the generalized- α method. *Trans. ASME J. Appl. Mech.* **60**(2), 371–375 (1993)
25. Cuadrado, J., Cardenal, J., Morer, P., Bayo, E.: Intelligent simulation of multibody dynamics: space-state and descriptor methods in sequential and parallel computing environments. *Multibody Syst. Dyn.* **4**, 55–73 (2000)
26. Daniel, W.J.T.: Analysis and implementation of a new constant acceleration algorithm. *Int. J. Numer. Methods. Eng.* **40**, 2841–2855 (1997)
27. Datar, M., Stanciulescu, I., Negrut, D.: A co-simulation framework for full vehicle analysis. In: Proceedings of the SAE 2011 World Congress, SAE Technical Paper 2011-01-0516, 12–14 April, Detroit, Michigan, USA (2011)
28. Datar, M., Stanciulescu, I., Negrut, D.: A co-simulation environment for high-fidelity virtual prototyping of vehicle systems. *Int. J. Veh. Syst. Model. Test.* **7**, 54–72 (2012)
29. Dörfel, M.R., Simeon, B.: Analysis and acceleration of a fluid-structure interaction coupling scheme. In: *Numerical Mathematics and Advanced Applications*, pp. 307–315 (2010)
30. D’Silva, S., Sundaram, P., Ambrosio, J.: Co-simulation platform for diagnostic development of a controlled chassis system. In: SAE Technical Paper 2006-01-1058 (2006). <https://doi.org/10.4271/2006-01-1058>
31. Eberhard, P., Gaugele, T., Heisel, U., Storchak, M.: A discrete element material model used in a co-simulated charpy impact test and for heat transfer. In: Proceedings 1st International Conference on Process Machine Interactions, Hannover, Germany, 3–4 Sept (2008)
32. Fancello, M., Masarati, P., Morandini, M.: Adding non-smooth analysis capabilities to general-purpose multibody dynamics by co-simulation. In: Proceedings of the ASME 9th MSNDC, Portland, OR, USA, 4–7 Aug, DETC2013-12208 (2013)
33. Fancello, M., Morandini, M., Masarati, P.: Helicopter rotor sailing by non-smooth dynamics co-simulation. *Arch. Mech. Eng.* **61**(2), 253–268 (2014). <https://doi.org/10.2478/meceng-2014-0015>
34. Friedrich M., Ulbrich, H.: A parallel co-simulation for mechatronic systems. In: Proceedings of The 1st Joint International Conference on Multibody System Dynamics, IMSD 2010, Lappeenranta, Finland, 25–27 May (2010)
35. Garcia de Jalon, J., Bayo, E.: *Kinematic and Dynamic Simulation of Multibody Systems. The Real-Time Challenge*. Springer, New York (1994)
36. Gear, C.W., Wells, D.R.: Multirate linear multistep methods. *BIT* **24**, 484–502 (1984)
37. Gonzalez, F., Gonzalez, M., Cuadrado, J.: Weak coupling of multibody dynamics and block diagram simulation tools. In: Proceedings of the ASME 2009 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference, IDETC/CIE 2009, San Diego, California, USA, 30 Aug–2 Sept (2009)
38. Gonzalez, F., Gonzalez, M., Mikkola, A.: Efficient coupling of multibody software with numerical computing environments and block diagram simulators. *Multibody Syst. Dyn.* **24**(3), 237–253 (2010). <https://doi.org/10.1007/s11044-010-9199-6>
39. Gonzalez, F., Naya, M.A., Luaces, A., Gonzalez, M.: On the effect of multirate co-simulation techniques in the efficiency and accuracy of multibody system dynamics. *Multibody Syst. Dyn.* **25**(4), 461–483 (2011)
40. Gu, B., Asada, H.H.: Co-simulation of algebraically coupled dynamic subsystems without disclosure of proprietary subsystem models. *J. Dyn. Syst. Meas. Control* **126**, 1–13 (2004). <https://doi.org/10.1115/1.1648307>
41. Hairer, E., Norsett, S.P., Wanner, G.: *Solving Ordinary Differential Equations I: Nonstiff Problems*, 3rd edn. Springer, Berlin (2009)

42. Hairer, E., Wanner, G.: Solving Ordinary Differential Equations II: Stiff and Differential-Algebraic Problems, 2nd edn. Springer, Berlin (2010)
43. Helduser, S., Stuewing, M., Liebig, S., Dronka, S.: Development of electro-hydraulic actuators using linked simulation and hardware-in-the-loop technology. In: Proceedings of Symposium on Power Transmission and Motion Control 2001, PTMC 2001, Bath, UK, 15–17 Sept (2001)
44. Hippmann, G., Arnold, M., Schittenhelm, M.: Efficient simulation of bush and roller chain drives. In: Goicolea, J., Cuadrado, J., Orden, J.G. (eds.) Proceedings of ECCOMAS Thematic Conference on Advances in Computational Multibody Dynamics, Madrid, pp. 1–18 (2005)
45. Kübler, R., Schiehlen, W.: Two methods of simulator coupling. *Math. Comput. Model. Dyn. Syst.* **6**, 93–113 (2000)
46. Lacoursiere, C., Nordfeldth, F., Linde, M.: A partitioning method for parallelization of large systems in realtime. In: Proceedings of the 3rd Joint International Conference on Multibody System Dynamics and The 7th Asian Conference on Multibody Dynamics, IMSD 2014, ACMD 2014, Bexco, Busan, Korea, 30 June–3 July (2014)
47. Lehnart, A., Fleissner, F., Eberhard, P.: Using SPH in a co-simulation approach to simulate sloshing in tank vehicles. In: Proceedings SPHERIC4, Nantes, France, 27–29 May (2009)
48. Leyendecker, S., Betsch, P., Steinmann, P.: The discrete null space method for the energy consistent integration of constrained mechanical systems. Part III: flexible multibody dynamics. *Multibody Syst. Dyn.* **19**(1–2), 45–72 (2008)
49. Leyendecker, S., Betsch, P., Steinmann, P.: Objective energy-momentum conserving integration for the constrained dynamics of geometrically exact beams. *Comput. Methods Appl. Mech. Eng.* **195**, 2313–2333 (2006)
50. Leyendecker, S., Betsch, P., Steinmann, P.: Energy-conserving integration of constrained Hamiltonian systems—a comparison of approaches. *Comput. Mech.* **33**(3), 174–185 (2004)
51. Li, P., Lu, D., Schweizer, B.: On the stability of explicit and implicit co-simulation approaches. In: ECCOMAS Thematic Conference on Multibody Dynamics, 29 June–2 July, Barcelona, Catalonia, Spain (2015)
52. Liao, Y.G., Du, H.I.: Co-simulation of multi-body-based vehicle dynamics and an electric power steering control system. *Proc. Inst. Mech. Eng. K J. Multibody Dyn.* **215**, 141–151 (2001)
53. Lu, D., Li, P., Schweizer, B.: Index-2 co-simulation approach for solver coupling with algebraic constraints. In: ECCOMAS thematic conference on multibody dynamics, June 29–July 2, Barcelona, Catalonia, Spain (2015)
54. Malczyk, P., Fraczek, J.: Evaluation of parallel efficiency in modeling of mechanisms using commercial multibody solvers. *Arch. Mech. Eng.* **LVI**(3), 237–249 (2009)
55. Meynen, S., Mayer, J., Schäfer, M.: Coupling algorithms for the numerical simulation of fluid-structure-interaction problems. In: ECCOMAS 2000: European Congress on Computational Methods in Applied Sciences and Engineering, Barcelona (2000)
56. Miao, J.C., Zhu, P., Shi, G.L., Chen, G.L.: Study on sub-cycling algorithm for flexible multi-body system—integral theory and implementation flow chart. *Comput. Mech.* **41**, 257–268 (2008). <https://doi.org/10.1007/s00466-007-0183-9>
57. Naya, M., Cuadrado, J., Dopico, D., Lugris, U.: An efficient unified method for the combined simulation of multibody and hydraulic dynamics: comparison with simplified and co-integration approaches. *Arch. Mech. Eng.* **LVIII**, 223–243 (2011)
58. Neal, M.O., Belytschko, T.: Explicit-explicit subcycling with non-integer time step ratios for structural dynamic systems. *Comput. Struct.* **31**, 871–880 (1989)
59. Negrut, D., Rampalli, R., Ottarsson, G.: On an implementation of the HHT method in the context of index 3 differential algebraic equations of multibody dynamics. *ASME J. Comput. Nonlinear Dyn.* **2**, 73–85 (2007)
60. Negrut, D., Tasora, A., Mazhar, H., Heyn, T., Hahn, P.: Leveraging parallel computing in multibody dynamics. *Multibody Syst. Dyn.* **27**, 95–117 (2012). <https://doi.org/10.1007/s11044-011-9262-y>
61. Negrut, D., Melanz, D., Mazhar, H., Lamb, D., Jayakumar, P.: Investigating through simulation the mobility of light tracked vehicles operating on discrete granular terrain. *SAE Int. J. Passeng. Cars Mech. Syst.* **6**, 369–381 (2013). <https://doi.org/10.4271/2013-01-1191>

62. Negrut, D., Serban, R., Mazhar, H., Heyn, T.: Parallel computing in multibody system dynamics: why, when and how. *J. Comput. Nonlinear Dyn.* **9**(4), 041007 (2014). <https://doi.org/10.1115/1.4027313>
63. Quaranta, G., Masarati, P., Mantegazza, P.: Multibody analysis of controlled aeroelastic systems on parallel computers. *Multibody Syst. Dyn.* **8**(1), 71–102 (2002). <https://doi.org/10.1023/A:1015894729968>
64. Shabana, A.A.: *Dynamics of Multibody Systems*. Cambridge University Press, Cambridge (2005)
65. Schäfer, M., Yigit, S., Heck, M.: Implicit partitioned fluid-structure interaction coupling. In: ASME, PVP2006-ICPVT11-93184, Vancouver, Canada (2006)
66. Schiehlen, W.: Multibody system dynamics: roots and perspectives. *Multibody Syst. Dyn.* **1**, 149–188 (1997)
67. Schmoll, R., Schweizer, B.: Co-simulation of multibody and hydraulic systems: comparison of different coupling approaches. In: Samin, J.C., Fiset, P. (eds.) *Multibody Dynamics 2011*, ECCOMAS Thematic Conference, Brussels, Belgium, 4–7 July 2011, pp. 1–13
68. Schweizer, B., Lu, D.: Predictor/corrector co-simulation approaches for solver coupling with algebraic constraints. *ZAMM J. Appl. Math. Mech.* (2014) <https://doi.org/10.1002/zamm.201300191>
69. Schweizer, B., Lu, D.: Semi-Implicit co-simulation approach for solver coupling. *Arch. Appl. Mech.* (2014). <https://doi.org/10.1007/s00419-014-0883-5>
70. Schweizer, B., Lu, D.: Stabilized index-2 co-simulation approach for solver coupling with algebraic constraints. *Multibody Syst. Dyn.* (2014). <https://doi.org/10.1007/s11044-014-9422-y>
71. Schweizer, B., Li, P., Lu, D.: Explicit and implicit co-simulation methods: stability and convergence analysis for different solver coupling approaches. *J. Comput. Nonlinear Dyn.* (2014). <https://doi.org/10.1115/1.4028503>
72. Schweizer, B., Li, P., Lu, D.: Implicit co-simulation methods: stability and convergence analysis for solver coupling with algebraic constraints. *ZAMM J. Appl. Math. Mech.* (2015). <https://doi.org/10.1002/zamm.201400087>
73. Schweizer, B., Li, P., Lu, D., Meyer, T.: Stabilized implicit co-simulation method: solver coupling with algebraic constraints for multibody systems. *J. Comput. Nonlinear Dyn.* (2015). <https://doi.org/10.1115/1.4030508>
74. Schweizer, B., Li, P., Lu, D., Meyer, T.: Stabilized implicit co-simulation methods: solver coupling based on constitutive laws. *Arch. Appl. Mech.* (2015). <https://doi.org/10.1007/s00419-015-0999-2>
75. Schweizer, B., Lu, D., Li, P.: Co-simulation method for solver coupling with algebraic constraints incorporating relaxation techniques. *Multibody Syst. Dyn.* (2015). <https://doi.org/10.1007/s11044-015-9464-9>
76. Serban, R., Melanz, D., Li, A., Stanciulescu, I., Jayakumar, P., Negrut, D.: A GPU-based preconditioned Newton-Krylov solver for flexible multibody dynamics. *Int. J. Numer. Meth. Eng.* **102**(9), 1585–1604 (2015)
77. Sicklinger, S., Belsky, V., Engelmann, B., Elmqvist, H., Olsson, H., Wüchner, R., Bletzinger, K.-U.: Interface Jacobian-based co-simulation. *Int. J. Numer. Meth. Eng.* **98**, 418–444 (2014). <https://doi.org/10.1002/nme.4637>
78. Simeon, B.: *Computational Flexible Multibody Dynamics: A Differential-Algebraic Approach*. Springer, Heidelberg (2013)
79. Spreng, F., Eberhard, P., Fleissner, F.: An approach for the coupled simulation of machining processes using multibody system and smoothed particle hydrodynamics algorithms. *Theor. Appl. Mech. Lett.* **3**(1), 8–013005 (2013)
80. Solcia, T., Masarati, P.: Efficient multirate simulation of complex multibody systems based on free software. In: *Proceedings of the ASME IDETC/CIE 2011*, 28–31 August, Washington, DC, USA, DETC2011-47306 (2011)
81. Tomulik, P., Fraczek, J.: Simulation of multibody systems with the use of coupling techniques: a case study. *Multibody Syst. Dyn.* **25**(2), 145–165 (2011)

82. Verhoeven, A., Tasic, B., Beelen, T.G.J., ter Maten, E.J.W., Mattheij, R.M.M.: BDF compound-fast multirate transient analysis with adaptive stepsize control. *J. Numer. Anal. Ind. Appl. Math.* **3**(3–4), 275–297 (2008)
83. Wang, J., Ma, Z.D., Hulbert, G.: A gluing algorithm for distributed simulation of multibody systems. *Nonlinear Dyn.* **34**, 159–188 (2003)
84. Wuensche, S., Clauß, C., Schwarz, P., Winkler, F.: Electro-thermal circuit simulation using simulator coupling. *IEEE Trans. Very Large Scale Integr. Syst.* **5**, 277–282 (1997)
85. Zierath, J., Woernle, C.: Development of a Dirichlet-to-Neumann algorithm for contact analysis in boundary element systems and its application to MBS-BEM co-simulation. In: Samin, J.C., Fiset, P. (eds.) *Multibody Dynamics 2011, ECCOMAS Thematic Conference*, Brussels, Belgium, 4–7 July 2011
86. Zierath, J., Woernle, C.: Contact modelling in multibody systems by means of a boundary element co-simulation and a Dirichlet-to-Neumann algorithm. In: Oñate, E. (series ed.) *Computational Methods in Applied Sciences*. Springer, Berlin (2012)

Chapter 10

The Influence of Secondary Flow on the Dynamics of Vibrating Tubes



J. P. Meijaard

Abstract The secondary flow inside a round tube undergoing motion perpendicular to its longitudinal axis is considered. The motion of the fluid is modelled by a discretization of the secondary flow and is simulated separately from the motion of the tube, which is modelled as a flexible multibody system. The interaction between the two problems is taken into account by means of co-simulation. The Navier–Stokes equations are simplified and linearized, and then discretized with a spectral expansion in the circumferential direction and with finite elements in the radial direction. A tube rotating with a constant angular velocity and a tube undergoing a harmonic vibration are considered as examples. The main objective of this study is to show that the proposed co-simulation is actually feasible for this problem.

10.1 Introduction

This contribution considers the simulation of the motion of a rigid or flexible tube conveying a fluid. The cross-section of the enclosed volume has a circular shape. For a stationary straight tube, the flow, if it is laminar below a critical Reynolds number, can be described by a Poiseuille–Hagen flow with a parallel flow in the direction of the centre line of the tube having a parabolic velocity profile. For a fully developed turbulent flow, the average velocity profile is almost a constant with a small boundary layer near the tube wall.

If the tube is subjected to a motion, linear accelerations of the tube can be transferred to the fluid by a hydrostatic pressure. On the other hand, if the tube rotates, the non-uniform velocity profile causes a non-uniform Coriolis term, which results in secondary flow in the tube, that is, a perturbation on the parabolic velocity distribution of the flow along the axis of the tube and additional flow in the lateral directions.

J. P. Meijaard (✉)
Olton Engineering Consultancy, Enschede, The Netherlands
e-mail: J.P.Meijaard@olton.nl

For low Reynolds numbers, the viscosity of the fluid suppresses the secondary flow, whereas for high Reynolds numbers, the flow is turbulent and the average flow velocity is almost uniform over the cross-section of the tube, so the Coriolis term is nearly constant on average and has little influence on the average flow. In an interval of Reynolds numbers just below the transition to turbulent flow, the secondary flow may become noticeable in the dynamics of the tube.

Although the secondary flow remains small in most cases, it can become important in high-precision instruments, in particular in Coriolis mass flow meters. Indeed, experiments have shown a deviation from the theoretical results based on the primary flow only for the sensitivity just before the transition to turbulent flow [1].

The main interest is the linear response of the tube, so the fluid flow equations can be linearized around the primary flow. The discretization is performed by a spectral expansion around the circumferential direction and a one-dimensional discretization along the radial direction. The changes along the length of the tube are assumed to be an order of magnitude smaller than those along the radial and circumferential directions and may therefore be neglected, so for each considered section along the length of the tube, the flow may be assumed to be independent of the coordinate along the length of the tube, but it may differ from section to section.

The model and the calculations for the fluid flow are coupled to a program for simulating flexible multibody dynamics systems [2]. The difficulty of the coupling is in the unknown accelerations and forces that have to be communicated between the two models. The fluid flow depends on the angular acceleration of the tube, besides the angular velocity, whereas the acceleration of the tube depends on the forces generated by the fluid flow. The contribution of the primary flow can be treated separately from the contributions of the secondary flow and be directly included in the multibody system model [3], so only the secondary flow needs to be considered in the co-simulation. The unknown accelerations are obtained from the numerical differentiation of the velocities based on their values over some previous time-steps, whereas the resulting forces can be directly returned to the multibody system model. The numerical integration is done with a standard third-order Runge–Kutta method for the multibody system and a second-order Runge–Kutta method (Heun’s method) with a smaller step size for the fluid flow.

Two examples are considered. Firstly, the fluid motion in a rigid tube with a steady rotation is considered. Secondly, a sinusoidal variation of the rotation is added.

10.2 Fluid Flow

In this section, the equations for the fluid flow, the Navier–Stokes equations, are simplified, an analytic solution for the primary flow and the linearized secondary flow if the angular velocity of the tube is constant are derived and a discretization scheme is proposed.

10.2.1 Simplified Equations for the Fluid Flow and an Analytical Solution for Constant Angular Velocity

The fluid flow is described by the Navier–Stokes equations for incompressible flow [4],

$$\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla) \mathbf{u} = -\frac{1}{\rho} \nabla p + \nu \Delta \mathbf{u} + g, \quad \nabla \cdot \mathbf{u} = 0. \quad (10.1)$$

Here, \mathbf{u} is the velocity vector relative to the moving coordinate system, p is the pressure, positive for compression, ρ is the fluid density, ν is the kinematic viscosity, ∇ is the gradient operator, $\Delta = \nabla \cdot \nabla$ is the Laplacian operator and g represents the forcing due to gravity and the moving coordinate system, including the Coriolis and centripetal terms. In cylindrical coordinates with longitudinal coordinate x along the length of the tube, the radius r measured from the centre line of the tube and the circumferential angle φ , and written out in components, where u_x , u_r and u_φ denote the physical components of the relative velocity, these equations become

$$\begin{aligned} \frac{\partial u_x}{\partial t} + u_x \frac{\partial u_x}{\partial x} + u_r \frac{\partial u_x}{\partial r} + \frac{u_\varphi}{r} \frac{\partial u_x}{\partial \varphi} \\ = -\frac{1}{\rho} \frac{\partial p}{\partial x} + \nu \left[\frac{\partial^2 u_x}{\partial x^2} + \frac{\partial^2 u_x}{\partial r^2} + \frac{1}{r^2} \frac{\partial^2 u_x}{\partial \varphi^2} + \frac{1}{r} \frac{\partial u_x}{\partial r} \right] + g_x, \end{aligned} \quad (10.2)$$

$$\begin{aligned} \frac{\partial u_r}{\partial t} + u_x \frac{\partial u_r}{\partial x} + u_r \frac{\partial u_r}{\partial r} + \frac{u_\varphi}{r} \left(\frac{\partial u_r}{\partial \varphi} - u_\varphi \right) \\ = -\frac{1}{\rho} \frac{\partial p}{\partial r} + \nu \left[\frac{\partial^2 u_r}{\partial x^2} + \frac{\partial^2 u_r}{\partial r^2} + \frac{1}{r^2} \frac{\partial^2 u_r}{\partial \varphi^2} + \frac{1}{r} \frac{\partial u_r}{\partial r} - \frac{2}{r^2} \frac{\partial u_\varphi}{\partial \varphi} - \frac{u_r}{r^2} \right] + g_r, \end{aligned} \quad (10.3)$$

$$\begin{aligned} \frac{\partial u_\varphi}{\partial t} + u_x \frac{\partial u_\varphi}{\partial x} + u_r \frac{\partial u_\varphi}{\partial r} + \frac{u_\varphi}{r} \left(\frac{\partial u_\varphi}{\partial \varphi} + u_r \right) \\ = -\frac{1}{\rho r} \frac{\partial p}{\partial \varphi} + \nu \left[\frac{\partial^2 u_\varphi}{\partial x^2} + \frac{\partial^2 u_\varphi}{\partial r^2} + \frac{1}{r^2} \frac{\partial^2 u_\varphi}{\partial \varphi^2} + \frac{1}{r} \frac{\partial u_\varphi}{\partial r} + \frac{2}{r^2} \frac{\partial u_r}{\partial \varphi} - \frac{u_\varphi}{r^2} \right] + g_\varphi, \end{aligned} \quad (10.4)$$

$$\frac{\partial u_x}{\partial x} + \frac{\partial u_r}{\partial r} + \frac{u_r}{r} + \frac{1}{r} \frac{\partial u_\varphi}{\partial \varphi} = 0. \quad (10.5)$$

For the case of a stationary tube of infinite length with no forcing, $g_x = g_r = g_\varphi = 0$, these equations admit an exact solution, the Poiseuille–Hagen flow,

$$u_x = -\frac{\partial p}{\partial x} \cdot \frac{1}{4\rho\nu} \cdot (r_i^2 - r^2) = u_0(1 - r^2/r_i^2), \quad u_r = 0, \quad u_\varphi = 0, \quad (10.6)$$

where r_i is the inner radius of the tube, $\partial p/\partial x$ is a constant, $p = (x - x_0)(\partial p/\partial x)$ and $u_0 = -(\partial p/\partial x)r_i^2/(4\rho\nu)$.

For a more general case, we assume that the tube undergoes a motion in the local xz -plane, that is, along directions with $\varphi = \pi/2$ or $\varphi = 3\pi/2$. This means that the acceleration is determined by linear accelerations in the local x - and z -directions, a_x and a_z , and the angular velocity and angular acceleration about a line parallel to the y -direction, $\dot{\omega}_y$ and $\ddot{\omega}_y$. The resulting acceleration forcing in Cartesian components is

$$\begin{bmatrix} g_x \\ g_y \\ g_z \end{bmatrix} = \begin{bmatrix} -a_x - z\dot{\omega}_y + x\omega_y^2 - 2\omega_y u_z \\ 0 \\ -a_z + x\dot{\omega}_y + z\omega_y^2 + 2\omega_y u_x \end{bmatrix}. \quad (10.7)$$

A part of these forcing terms can be taken up by a hydrostatic pressure distribution,

$$p_0 = \rho \left[\frac{1}{2}\omega_y^2(x^2 + z^2) - a_x x - a_z z + \dot{\omega}_y x z \right]. \quad (10.8)$$

The forcing that is not compensated by this hydrostatic pressure is given by

$$\begin{bmatrix} g_x \\ g_r \\ g_\varphi \end{bmatrix} = \begin{bmatrix} -2\dot{\omega}_y r \sin \varphi - 2\omega_y(u_r \sin \varphi + u_\varphi \cos \varphi) \\ 2\omega_y u_x \sin \varphi \\ 2\omega_y u_x \cos \varphi \end{bmatrix}. \quad (10.9)$$

It is further assumed that the dependence of the flow on the longitudinal coordinate x is weak, so derivatives with respect to this coordinate can be neglected; furthermore, the flow is linearized around the Poiseuille–Hagen solution. These simplifications result in the equations

$$\frac{\partial u_x}{\partial t} - u_r u_0 \frac{2r}{r_i^2} = \nu \left[\frac{\partial^2 u_x}{\partial r^2} + \frac{1}{r^2} \frac{\partial^2 u_x}{\partial \varphi^2} + \frac{1}{r} \frac{\partial u_x}{\partial r} \right] - 2\dot{\omega}_y r \sin \varphi, \quad (10.10)$$

$$\begin{aligned} \frac{\partial u_r}{\partial t} = & -\frac{1}{\rho} \frac{\partial p}{\partial r} + \nu \left[\frac{\partial^2 u_r}{\partial r^2} + \frac{1}{r^2} \frac{\partial^2 u_r}{\partial \varphi^2} + \frac{1}{r} \frac{\partial u_r}{\partial r} - \frac{2}{r^2} \frac{\partial u_\varphi}{\partial \varphi} - \frac{u_r}{r^2} \right] \\ & + 2\omega_y u_0 \left(1 - \frac{r^2}{r_i^2} \right) \sin \varphi, \end{aligned} \quad (10.11)$$

$$\begin{aligned} \frac{\partial u_\varphi}{\partial t} = & -\frac{1}{\rho r} \frac{\partial p}{\partial \varphi} + \nu \left[\frac{\partial^2 u_\varphi}{\partial r^2} + \frac{1}{r^2} \frac{\partial^2 u_\varphi}{\partial \varphi^2} + \frac{1}{r} \frac{\partial u_\varphi}{\partial r} + \frac{2}{r^2} \frac{\partial u_r}{\partial \varphi} - \frac{u_\varphi}{r^2} \right] \\ & + 2\omega_y u_0 \left(1 - \frac{r^2}{r_i^2} \right) \cos \varphi, \end{aligned} \quad (10.12)$$

$$\frac{\partial u_r}{\partial r} + \frac{u_r}{r} + \frac{1}{r} \frac{\partial u_\varphi}{\partial \varphi} = 0. \quad (10.13)$$

Note that u_x and p now have the meaning of perturbations of the longitudinal component of the velocity and of the pressure with respect to the Poiseuille–Hagen flow.

These linearized equations can still be further simplified by noting that the forcing depends on the angle φ only in the first harmonic, so solutions can be found in the form

$$\begin{aligned} u_x &= u_{x1}(r, t) \sin \varphi, \\ u_r &= u_{r1}(r, t) \sin \varphi, \\ u_\varphi &= u_{\varphi1}(r, t) \cos \varphi, \\ p &= p_1(r, t) \sin \varphi. \end{aligned} \quad (10.14)$$

Substituting these expressions into (10.10)–(10.13) yields

$$\frac{\partial u_{x1}}{\partial t} - u_{r1} u_0 \frac{2r}{r_i^2} = \nu \left[\frac{\partial^2 u_{x1}}{\partial r^2} + \frac{1}{r} \frac{\partial u_{x1}}{\partial r} - \frac{u_{x1}}{r^2} \right] - 2\omega_y r, \quad (10.15)$$

$$\frac{\partial u_{r1}}{\partial t} = -\frac{1}{\rho} \frac{\partial p_1}{\partial r} + \nu \left[\frac{\partial^2 u_{r1}}{\partial r^2} + \frac{1}{r} \frac{\partial u_{r1}}{\partial r} - \frac{2u_{r1}}{r^2} + \frac{2u_{\varphi1}}{r^2} \right] + 2\omega_y u_0 \left(1 - \frac{r^2}{r_i^2} \right), \quad (10.16)$$

$$\frac{\partial u_{\varphi1}}{\partial t} = -\frac{p_1}{\rho r} + \nu \left[\frac{\partial^2 u_{\varphi1}}{\partial r^2} + \frac{1}{r} \frac{\partial u_{\varphi1}}{\partial r} - \frac{2u_{\varphi1}}{r^2} + \frac{2u_{r1}}{r^2} \right] + 2\omega_y u_0 \left(1 - \frac{r^2}{r_i^2} \right), \quad (10.17)$$

$$u_{\varphi1} = r \frac{\partial u_{r1}}{\partial r} + u_{r1}. \quad (10.18)$$

These equations have the property that (10.16)–(10.18) can be solved first, as these equations do not depend on u_{x1} , and then the solution can be used to solve (10.15).

For the case of a constant angular velocity ω_y , an exact stationary solution of the linearized equations can be found as [5]

$$u_{r1} = \frac{\omega_y u_0}{48\nu r_i^2} (r_i^2 - r^2)^2, \quad u_{\varphi1} = \frac{\omega_y u_0}{48\nu r_i^2} (r_i^2 - r^2) (r_i^2 - 5r^2) \quad (10.19)$$

and

$$u_{x1} = \frac{\omega_y u_0^2}{1152\nu^2 r_i^4} (3r_i^6 r - 6r_i^4 r^3 + 4r_i^2 r^5 - r^7), \quad p_1 = \frac{\rho \omega_y u_0}{6r_i^2} (10r_i^2 r - 3r^3). \quad (10.20)$$

The forces of the fluid on the wall of the tube consist of a pressure p normal to the tube wall and two components of the shearing force,

$$\tau_{xr} = \rho\nu \frac{\partial u_x}{\partial r}, \quad \tau_{r\varphi} = \rho\nu \frac{\partial u_\varphi}{\partial r}. \quad (10.21)$$

For the case that ω_y is constant, the pressure and the shear stresses due to the secondary flow are

$$p = \frac{7}{6}\rho\omega_y u_0 r_i \sin \varphi, \quad \tau_{xr} = -\frac{\rho\omega_y u_0^2 r_i^2}{576\nu} \sin \varphi, \quad \tau_{r\varphi} = \frac{1}{6}\rho\omega_y u_0 r_i \cos \varphi. \quad (10.22)$$

There is no resulting additional longitudinal force and a resultant lateral force per unit of length equal to

$$f_z = \pi\rho\omega_y u_0 r_i^2, \quad (10.23)$$

which is equal to the Coriolis force on the fluid. There is a resulting moment about the y -axis per unit of length,

$$m_y = \frac{\pi\rho\omega_y u_0^2 r_i^4}{576\nu}. \quad (10.24)$$

Note that the hydrostatic pressure distribution in (10.8) yields no nett force on the tube in this case.

10.2.2 Discretization Scheme

For the more general case with variable rate of rotation, or if non-linear terms are included, no general analytic solutions can be found. Therefore, a numerical analysis is performed. The radial flow field u_{r1} is discretized by a C^1 continuous approximation by piecewise cubic Hermite polynomials. On an interval $r_p \leq r \leq r_q$ of length $l = r_q - r_p$, the interpolation is

$$u_{r1}(r, t) = u_{r1p}(t) (1 - 3\xi^2 + 2\xi^3) + lu'_{r1p}(t) (\xi - 2\xi^2 + \xi^3) + u_{r1q}(t) (3\xi^2 - 2\xi^3) + lu'_{r1q}(t) (-\xi^2 + \xi^3). \quad (10.25)$$

Here, u_{r1p} , u_{r1q} , u'_{r1p} and u'_{r1q} are time-dependent coefficients representing the velocities and their radial derivatives at the end-points of the interval and $\xi = (r - r_p)/l$, $0 \leq \xi \leq 1$ is the dimensionless coordinate. The circumferential velocity distribution follows from the incompressibility condition (10.18). The axial flow is discretized in the same way as the radial flow,

$$u_{x1}(r, t) = u_{x1p}(t) (1 - 3\xi^2 + 2\xi^3) + lu'_{x1p}(t) (\xi - 2\xi^2 + \xi^3) + u_{x1q}(t) (3\xi^2 - 2\xi^3) + lu'_{x1q}(t) (-\xi^2 + \xi^3). \quad (10.26)$$

The dissipation rate per unit of volume is given by

$$\rho\nu (2\dot{\varepsilon}_x^2 + 2\dot{\varepsilon}_r^2 + 2\dot{\varepsilon}_\varphi^2 + \dot{\gamma}_{xr}^2 + \dot{\gamma}_{x\varphi}^2 + \dot{\gamma}_{r\varphi}^2), \quad (10.27)$$

where the strain rates are given by

$$\dot{\epsilon}_x = \frac{\partial u_x}{\partial x}, \quad \dot{\epsilon}_r = \frac{\partial u_r}{\partial r}, \quad \dot{\epsilon}_\varphi = \frac{1}{r} \frac{\partial u_\varphi}{\partial \varphi} + \frac{u_r}{r} \quad (10.28)$$

and the shear strain rates by

$$\dot{\gamma}_{xr} = \frac{\partial u_x}{\partial r} + \frac{\partial u_r}{\partial x}, \quad \dot{\gamma}_{x\varphi} = \frac{1}{r} \frac{\partial u_x}{\partial \varphi} + \frac{\partial u_\varphi}{\partial x}, \quad \dot{\gamma}_{r\varphi} = \frac{1}{r} \frac{\partial u_r}{\partial \varphi} + \frac{\partial u_\varphi}{\partial r} - \frac{u_\varphi}{r}. \quad (10.29)$$

With the expansion (10.14), where all derivatives with respect to x are zero and the incompressibility condition (10.18), integrating the dissipation over the surface of the tube yields the dissipation per unit of length of the tube as

$$\pi \rho \nu \int_0^{r_1} \left[4 \left(\frac{\partial u_{r1}}{\partial r} \right)^2 + \left(\frac{\partial u_{x1}}{\partial r} \right)^2 + \frac{u_{x1}^2}{r^2} + \left(r \frac{\partial^2 u_{r1}}{\partial r^2} + \frac{\partial u_{r1}}{\partial r} \right)^2 \right] r dr. \quad (10.30)$$

The element damping matrix for the nodal variables $[u_{r1p}, u'_{r1p}, u_{r1q}, u'_{r1q}]$ leaving out the common factor $\pi \rho$, is

$$\mathbf{D}_r = \frac{\nu r_p^3}{l^3} \begin{bmatrix} 12 & 6 & -12 & 6 \\ 6 & 4 & -6 & 2 \\ -12 & -6 & 12 & -6 \\ 6 & 2 & -6 & 4 \end{bmatrix} + \frac{\nu r_p^2}{l^2} \begin{bmatrix} 18 & 6 & -18 & 12 \\ 6 & 2 & -6 & 3 \\ -18 & -6 & 18 & -12 \\ 12 & 3 & -12 & 10 \end{bmatrix} \\ + \frac{\nu r_p}{l} \begin{bmatrix} 18 & 9/2 & -18 & 21/2 \\ 9/2 & 2 & -9/2 & 5/2 \\ -18 & -9/2 & 18 & -21/2 \\ 21/2 & 5/2 & -21/2 & 10 \end{bmatrix} + \nu \begin{bmatrix} 6 & 3/2 & -6 & 3 \\ 3/2 & 1/2 & -3/2 & 3/4 \\ -6 & -3/2 & 6 & -3 \\ 3 & 3/4 & -3 & 7/2 \end{bmatrix}. \quad (10.31)$$

The mass matrix is found from the expression for the kinetic energy per unit of length,

$$\pi \rho \int_0^{r_1} (\dot{u}_{x1}^2 + \dot{u}_{r1}^2 + \dot{u}_{\varphi 1}^2) r dr \\ = \pi \rho \int_0^{r_1} \left[\dot{u}_{x1}^2 + 2\dot{u}_{r1}^2 + 2r\dot{u}_{r1} \frac{\partial \dot{u}_{r1}}{\partial r} + r^2 \left(\frac{\partial \dot{u}_{r1}}{\partial r} \right)^2 \right] r dr. \quad (10.32)$$

For the in-plane flow, this leads to the mass matrix, where again the common factor $\pi \rho$ has been left out,

$$\begin{aligned}
\mathbf{M}_r = & \frac{r_p^3}{l} \begin{bmatrix} 6/5 & 1/10 & -6/5 & 1/10 \\ 1/10 & 2/15 & -1/10 & -1/30 \\ -6/5 & -1/10 & 6/5 & -1/10 \\ 1/10 & -1/30 & -1/10 & 2/15 \end{bmatrix} + r_p^2 \begin{bmatrix} 4/5 & 3/10 & -9/5 & 0 \\ 3/10 & 1/10 & -3/10 & -1/20 \\ -9/5 & -3/10 & 14/5 & 0 \\ 0 & -1/20 & 0 & 3/10 \end{bmatrix} \\
& + r_p l \begin{bmatrix} 36/35 & 3/14 & -36/35 & -3/35 \\ 3/14 & 2/35 & -3/14 & -3/70 \\ -36/35 & -3/14 & 106/35 & 3/35 \\ -3/35 & -3/70 & 3/35 & 9/35 \end{bmatrix} + l^2 \begin{bmatrix} 3/14 & 1/20 & -3/14 & -1/28 \\ 1/20 & 11/840 & -1/20 & -11/840 \\ -3/14 & -1/20 & 17/14 & 1/28 \\ -1/28 & -11/840 & 1/28 & 13/168 \end{bmatrix}. \tag{10.33}
\end{aligned}$$

The force vector is found from the power of the forcing per unit of length,

$$2\pi\rho\omega_y u_0 \int_0^{r_i} \left[2u_{r1} \left(1 - \frac{r^2}{r_i^2} \right) + r \frac{\partial u_{r1}}{\partial r} \left(1 - \frac{r^2}{r_i^2} \right) \right] r dr \tag{10.34}$$

This gives the element force vector, again scaled with $\pi\rho$,

$$\begin{aligned}
\frac{\mathbf{F}_r}{\omega_y u_0} = & \frac{r_p^4}{r_i^2} \begin{bmatrix} 2 \\ 0 \\ -2 \\ 0 \end{bmatrix} + \frac{r_p^3 l}{r_i^2} \begin{bmatrix} 2 \\ 1/3 \\ -6 \\ -1/3 \end{bmatrix} + \frac{r_p^2 l^2}{r_i^2} \begin{bmatrix} 9/5 \\ 2/5 \\ -39/5 \\ -3/5 \end{bmatrix} + \frac{r_p l^3}{r_i^2} \begin{bmatrix} 4/5 \\ 1/5 \\ -24/5 \\ -2/5 \end{bmatrix} \\
& + \frac{l^4}{r_i^2} \begin{bmatrix} 1/7 \\ 4/105 \\ -8/7 \\ -2/21 \end{bmatrix} + r_p^2 \begin{bmatrix} -2 \\ 0 \\ 2 \\ 0 \end{bmatrix} + r_p l \begin{bmatrix} 0 \\ 0 \\ 4 \\ 0 \end{bmatrix} + l^2 \begin{bmatrix} 0 \\ 0 \\ 2 \\ 0 \end{bmatrix}. \tag{10.35}
\end{aligned}$$

The system equations can be obtained with the familiar finite-element assembly process as

$$\mathbf{M}_f \dot{\mathbf{u}}_f + \mathbf{D}_f \mathbf{u}_f = \mathbf{F}_f + \mathbf{F}_{f0}, \tag{10.36}$$

where \mathbf{M}_f is the system mass matrix, \mathbf{D}_f is the system damping matrix, \mathbf{u}_f is the vector of system flow variables, \mathbf{F}_f is the system forcing vector and \mathbf{F}_{f0} is the system reaction force vector due to the boundary conditions at $r = r_i$. The resulting force on the tube per unit of length is given by $-\pi\rho F_{f0}$, where F_{f0} is the component of the reaction force vector corresponding to u_{r1} at $r = r_i$.

The perturbations of the axial flow can be modelled in the same way, but in this preliminary study, these and the resulting moment are neglected.

10.3 Coupling of the Fluid Model and the Multibody Dynamic Model

The mechanical system is modelled as a flexible multibody dynamics system with tube elements, which are flexible beams through which a fluid may flow [3]. The effects of the primary flow due to the hydrostatic pressure distribution (10.8) are

already included in this model, so the co-simulation only has to be applied for the secondary flow.

The equations of motion of the multibody system can be written as

$$\mathbf{M}_m(\mathbf{u}_m)\ddot{\mathbf{u}}_m + \mathbf{h}_m(\mathbf{u}_m, \dot{\mathbf{u}}_m, t) = \mathbf{F}_m, \quad (10.37)$$

where \mathbf{M}_m is the mass matrix of the multibody system, \mathbf{h}_m contains all forces, and \mathbf{F}_m are the fluid forces on the multibody system. Together with the fluid dynamics equations (10.36), these form the differential equations which determine the dynamic behaviour. The two subsystems are coupled: the fluid dynamics equations need the angular velocity of the tube and the multibody model needs the fluid dynamics forces on the tube. The numerical integration is done as follows. The multibody dynamics equations are rewritten as a system of first-order equations with the state variables $\mathbf{y} = (\mathbf{u}_m^T, \dot{\mathbf{u}}_m^T)^T$, and are integrated with a third-order explicit Runge–Kutta method, defined as

$$\begin{aligned} \mathbf{k}_1 &= \mathbf{f}(\mathbf{y}, t), \\ \mathbf{k}_2 &= \mathbf{f}(\mathbf{y} + h\mathbf{k}_1/3, t + h/3), \\ \mathbf{k}_3 &= \mathbf{f}(\mathbf{y} + 2h\mathbf{k}_2/3, t + 2h/3), \\ \mathbf{y}(t + h) &= \mathbf{y}(t) + h\mathbf{k}_1/4 + 3h\mathbf{k}_3/4, \end{aligned} \quad (10.38)$$

where h is the step size and

$$\mathbf{f}(\mathbf{y}, t) = \left[\begin{array}{c} \dot{\mathbf{u}}_m \\ \mathbf{M}_m^{-1}(\mathbf{F}_m - \mathbf{h}_m) \end{array} \right]. \quad (10.39)$$

At the function evaluations of \mathbf{f} , the solver for the fluid dynamics equations is called. As the evaluations are at equidistant time points one-third of the step size apart, the fluid dynamics equations can be solved with this smaller time step; a second-order explicit Runge–Kutta method, Heun's method, is used for this, which is defined as

$$\begin{aligned} \mathbf{l}_1 &= \mathbf{g}(\mathbf{u}_f, t), \\ \mathbf{l}_2 &= \mathbf{g}(\mathbf{u}_f + h\mathbf{l}_1/3, t + h/3), \\ \mathbf{u}_f(t + h/3) &= \mathbf{u}_f(t) + h(\mathbf{l}_1 + \mathbf{l}_2)/6, \end{aligned} \quad (10.40)$$

where

$$\mathbf{g}(\mathbf{u}_f, t) = \mathbf{M}_f^{-1}(\mathbf{F}_f - \mathbf{D}_f\mathbf{u}_f), \quad (10.41)$$

and the forces on the tube at specific sections are returned to the function \mathbf{f} . The coupling to the multibody system does not reduce the order of the second-order integrator, but the order of the third-order integrator is reduced to second order because of the coupling. This choice of the two different numerical integration methods is motivated by the fact the multibody system is mostly lightly damped, so the stability region of the integration method should include a part of the imaginary axis, as the third-order method does, but the fluid dynamics equations are of the first order and

mostly have eigenvalues with negative real parts, and they tend to be rather stiff if a small mesh size is chosen, so a smaller time step and a sufficiently large stability region along the negative real axis is needed, whereas the accuracy is less important. More details about integration methods can be found in [6, 7].

10.4 Test Results

A simple test system is proposed, viz a straight tube hinged at one side at which the fluid enters the tube and a free end at the other side from which the fluid exits the tube, see Fig. 10.1. Two test cases are considered: a spin-up motion and a harmonic vibration. The data for the system are listed in Table 10.1. The tube is modelled with two equal finite beam elements and the flow is discretized at the three nodal points with ten elements in the radial direction. Also cases in which the tube is rigid and the secondary flow is suppressed are considered as references.

As a first test, the spin-up of the tube from rest with a constant moment of $\pi \times 10^5$ Nm is considered. If the secondary flow would be absent, the angular veloc-

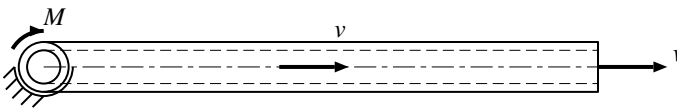


Fig. 10.1 A fluid-conveying tube

Table 10.1 Data for the test system

Description	Symbol	Value	Unit
Inner tube radius	r_i	0.0010	m
Outer tube radius	r_o	0.0015	m
Length of the tube	L	0.100	m
Mass density of the fluid	ρ	1000	kg/m ³
Kinematic viscosity of the fluid	ν	1.0×10^{-6}	m ² /s
Mass density of the tube	ρ_t	8000	kg/m ³
Young's modulus of the tube	E_t	2.0×10^{11}	N/m ²
Poisson's ratio of the tube	ν_t	0.3	—
Maximum axial fluid velocity	u_0	2.0	m/s
Reynolds number	Re	2000	—

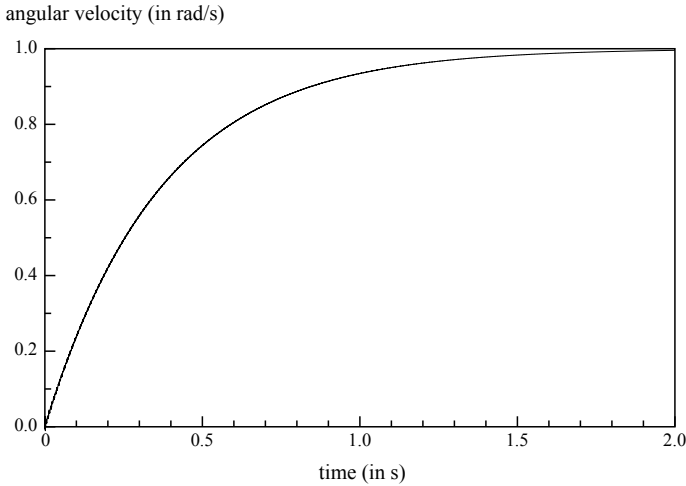


Fig. 10.2 Angular velocity at the base for the spin-up motion

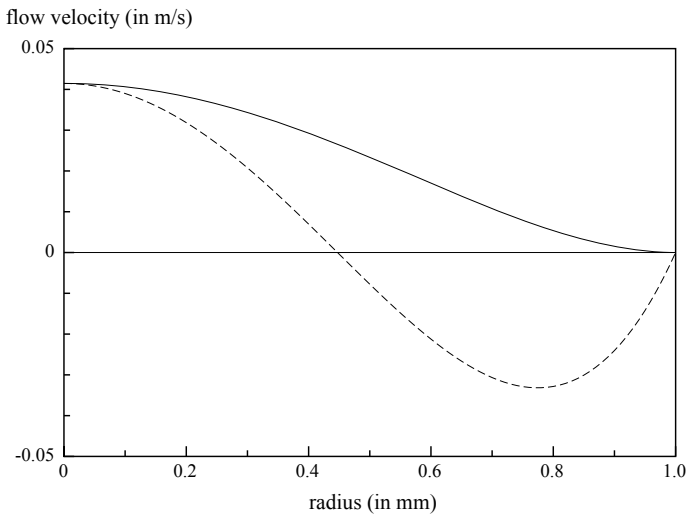


Fig. 10.3 Radial (full line) and circumferential (dashed line) fluid velocity distribution

ity would approach 1 rad/s. The increase in angular velocity at the base is shown in Fig. 10.2. There is almost no difference with the case in which the deflection of the tube and the secondary flow are neglected. The distributions of the radial flow velocity along the z -axis and the circumferential flow along the y -axis is shown in Fig. 10.3. The area under the curve of the circumferential flow is zero because of the incompressibility of the fluid. These curves agree very well with the theoretical solution (10.19).

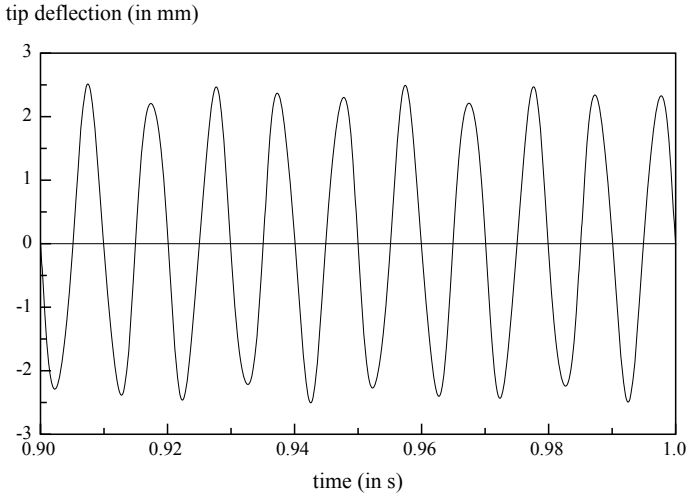


Fig. 10.4 Tip deflection in the z -direction of the tip of the tube when the base is subject to a harmonic prescribed motion

In a second test, the base of the tube is rotated harmonically with a frequency of 100 Hz and an amplitude of 0.1 rad. As the first natural frequency of the system is about 240 Hz, a considerable elastic deformation of the tube can be seen. Figure 10.4 shows the deflection of the tip for ten periods of the excitation from $t = 0.9$ s to $t = 1.0$ s. It can be seen that not all transients have faded away after hundred periods, which is partly due to the absence of material damping in the model for the tube vibrations. Neglecting the secondary flow now leads to differences of the order of magnitude of 0.1%.

10.5 Conclusions and Outlook

This initial study has shown that the co-simulation of the flexible multibody system with a fluid dynamics system of the flow inside a tube is feasible. The use of a third-order Runge–Kutta method for integrating the multibody dynamics equations in combination with a second-order method with a three times smaller step size for integrating the fluid dynamics equations proved to be convenient. The influence of the secondary flow on the total motion of the tube is quite small, but it can be significant for high-precision instruments, where it can lead to measurable differences and limit the accuracy if it is not compensated for.

The model can easily be extended to include axial flow; in that case, an extrapolation for the angular acceleration has to be made, because the forcing of the axial flow depends on this mechanical variable. Also non-linear terms can be included in

the analysis, for which the flow has also terms with higher harmonics in the circumferential direction.

Secondary flow can also be induced by the curvature of the tube, either because of its initial shape or because of the deformation of the tube. This leads to a quite similar analysis, which can be combined with the effect of the motion of the tube.

References

1. Mehendale, A.: Coriolis mass flow rate meters for low flows. Dissertation, University of Twente, Enschede (2008)
2. Jonker, J.B., Meijaard, J.P.: SPACAR—computer program for dynamic analysis of flexible spatial mechanisms and manipulators. In: Schiehlen, W. (ed.) *Multibody Systems Handbook*, pp. 123–143. Springer, Berlin (1990)
3. Meijaard, J.P.: Fluid-conveying flexible pipes modeled by large-deflection finite elements in multibody systems. *Trans. ASME J. Comput. Nonlinear Dyn.* **9**, 0110081–7 (2014)
4. Batchelor, G.K.: *An Introduction to Fluid Dynamics*. Cambridge University Press, Cambridge (1967)
5. Barua, S.N.: Secondary flow in a rotating straight pipe. *Proc. R. Soc. A* **227**, 133–139 (1954)
6. Hairer, E., Nørsett, S.P., Wanner, G.: *Solving Ordinary Differential Equations I, Nonstiff Problems (Second Revised Edition)*. Springer, Berlin (1993)
7. Hairer, E., Wanner, G.: *Solving Ordinary Differential Equations II, Stiff and Differential-Algebraic Problems (Second Revised Edition)*. Springer, Berlin (1996)

Chapter 11

Error Estimation Approach for Controlling the Communication Step-Size for Explicit Co-simulation Methods



Tobias Meyer, Jan Kraft, Daixing Lu and Bernhard Schweizer

Abstract In this paper, an approach for controlling the communication-step size in connection with explicit co-simulation methods is suggested. In the framework of the proposed communication-step size controller, each subsystem integration is carried out with two different explicit co-simulation methods. By comparing the variables for both integrations, an error estimator for the local error can be constructed. Making use of the estimated local error, a step-size controller for the communication step-size can be implemented. Examples are presented demonstrating the applicability and accuracy of the proposed communication-step size controller.

11.1 Introduction

The general idea of co-simulation is to separate a system of differential-algebraic equations into several subsystems. Since the numerical effort for time integration increases—often disproportionately high—with the number of state variables, integrating the subsystems in parallel is often more efficient than computing the overall monolithic system. Further, the subsystems may be computed with tailored subsystem solvers improving the efficiency. Co-simulation methods are classified with respect to the coupling technique. Applying a constraint coupling approach [5], the

T. Meyer (✉) · J. Kraft · D. Lu · B. Schweizer
Institute of Applied Dynamics, Technische Universität Darmstadt,
64287 Darmstadt, Germany
e-mail: meyer@ad.tu-darmstadt.de

J. Kraft
e-mail: kraft@ad.tu-darmstadt.de

D. Lu
e-mail: lu@ad.tu-darmstadt.de

B. Schweizer
e-mail: schweizer@ad.tu-darmstadt.de

subsystems are coupled by algebraic constraint equations. Using an applied force-coupling approach, the subsystems are coupled by constitutive laws [2]. This paper concentrates on applied-force coupling techniques.

The paper is organized as follows. In Sect. 11.2, coupling variables are introduced. With the help of a co-simulation test model, three different techniques for decomposing an overall system into subsystems are illustrated in Sect. 11.3. Next, two different explicit co-simulation approaches are explained. In Sect. 11.5, an error analysis for both methods is carried out. Then, the equations of motion of multibody subsystems, which are coupled by constitutive laws, are described in more detail. In Sect. 11.7, a co-simulation approach for controlling the communication-step size for coupled multibody systems is introduced. Finally, numerical examples are presented. In the appendix, another explicit co-simulation approach is proposed.

11.2 Definition of Coupling Variables

To keep the representation concise, the following notation will be used for transposed vectors. With $\mathbf{x}^1 = [x_1^1, \dots, x_{d_1}^1]^T, \dots, \mathbf{x}^n = [x_1^n, \dots, x_{d_n}^n]^T$, the vector $\mathbf{x} = [\mathbf{x}^1, \dots, \mathbf{x}^n]^T$ represents $\mathbf{x} = [x_1^1, \dots, x_{d_1}^1, \dots, x_1^n, \dots, x_{d_n}^n]^T$.

A multibody system is described by the equations of motion

$$\mathbf{B}(t, \mathbf{z}) \dot{\mathbf{z}} = \mathbf{F}(t, \mathbf{z}), \quad (11.1)$$

where the vector \mathbf{z} contains the position variables \mathbf{q} (displacement coordinates and rotation parameters), the velocity variables \mathbf{v} , and the reaction forces and torques $\boldsymbol{\lambda}$. The overall system (11.1) is partitioned into several subsystems. Making use of an applied-force coupling approach, the vector \mathbf{z} is decomposed into the vectors $[\mathbf{z}^1, \dots, \mathbf{z}^r]^T$, where the subsystem vector $\mathbf{z}^s = [\mathbf{q}^s, \mathbf{v}^s, \boldsymbol{\lambda}^s]^T$ contains the variables of subsystem $s = 1, \dots, r$. The matrix $\mathbf{B}(t, \mathbf{z})$ is assumed to be block diagonal. More precisely, there is one block for each subsystem, which is independent of the variables of the other subsystems, i.e. is to write $\mathbf{B}(t, \mathbf{z}) = \text{blockdiag}(\mathbf{B}^1(t, \mathbf{z}^1), \dots, \mathbf{B}^r(t, \mathbf{z}^r))$. Decomposing the right hand side of Eq. (11.1) accordingly yields $\mathbf{F}(t, \mathbf{z}) = [\mathbf{F}^1(t, \mathbf{z}), \dots, \mathbf{F}^r(t, \mathbf{z})]^T$. Thus the equations of motion of subsystem s read as

$$\mathbf{B}^s(t, \mathbf{z}^s) \dot{\mathbf{z}}^s = \mathbf{F}^s(t, \mathbf{z}) \quad (11.2)$$

for $s = 1, \dots, r$. Next, a vector $\mathbf{F}^{\text{co},s}(t, \mathbf{z}^s, \mathbf{u})$ and a vector of coupling functions $\mathbf{u}(t, \mathbf{z})$ are introduced, which are defined in such a way that the subsystem equations depend only implicitly on the state variables of the other subsystems. That means that the right hand side of subsystem s can be expressed as $\mathbf{F}^s(t, \mathbf{z}) = \mathbf{F}^{\text{co},s}(t, \mathbf{z}^s, \mathbf{u}(t, \mathbf{z}))$. The coupling functions $\mathbf{u}(t, \mathbf{z})$ are replaced by coupling variables \mathbf{p} , which are necessary to decompose the overall system into subsystems. Then, the equations of motion

of subsystem s are expressed as

$$\mathbf{B}^s(t, \mathbf{z}^s) \dot{\mathbf{z}}^s = \mathbf{F}^{\text{co},s}(t, \mathbf{z}^s, \mathbf{p}) \tag{11.3}$$

for $s = 1, \dots, r$. Arranging the right hand sides of all subsystems into the vector $\mathbf{F}^{\text{co}}(t, \mathbf{z}, \mathbf{p}) = [\mathbf{F}^{\text{co},1}(t, \mathbf{z}^1, \mathbf{p}), \dots, \mathbf{F}^{\text{co},r}(t, \mathbf{z}^r, \mathbf{p})]^T$ yields the decomposed system

$$\mathbf{B}(t, \mathbf{z}) \dot{\mathbf{z}} = \mathbf{F}^{\text{co}}(t, \mathbf{z}, \mathbf{p}), \tag{11.4}$$

$$\mathbf{0} = \mathbf{g}^{\text{co}}(t, \mathbf{z}, \mathbf{p}), \tag{11.5}$$

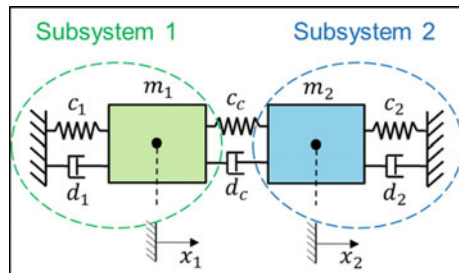
where the right hand side of the coupling condition is defined by $\mathbf{g}^{\text{co}}(t, \mathbf{z}, \mathbf{p}) := \mathbf{p} - \mathbf{u}(t, \mathbf{z})$.

Applying a co-simulation approach, a communication-time grid denoted by $T_0, \dots, T_N, \dots, T_{\text{end}}$ is introduced. The coupling conditions are only considered at the communication-time points, i.e. $\mathbf{g}^{\text{co}}(T_N, \mathbf{z}_N, \mathbf{p}(T_N)) = \mathbf{0}$ with $\mathbf{z}_N = \mathbf{z}(T_N)$. Approximating the coupling variables between two consecutive communication-time points T_N and T_{N+1} by a vector of polynomials, the subsystems can be integrated independently with individual subsystem solvers and individual subsystem step sizes. At the communication-time points, information is interchanged between the subsystems for updating the coupling variables.

11.3 Co-simulation Test Model

To illustrate the decomposition of an overall model into subsystems, the linear 2-DOF oscillator is considered, which is a well-established co-simulation test model. Figure 11.1 depicts a linear 2-mass oscillator (masses m_1, m_2 ; spring constants c_1, c_2 ; damping coefficients d_1, d_2 ; coupling-spring constant c_c ; coupling-damping coefficient d_c). The variables x_1 and x_2 denote the displacements of both masses; v_1 and v_2 describe the corresponding velocities. The springs are assumed to be stress-free for $x_1 = x_2 = 0$. The equations of motion of the overall system read as

Fig. 11.1 Linear 2-DOF oscillator: interpretation as two coupled single-mass oscillators



$$\begin{aligned}
\dot{x}_1 &= v_1, \\
m_1 \dot{v}_1 &= -c_1 x_1 - d_1 v_1 + c_c(x_2 - x_1) + d_c(v_2 - v_1), \\
\dot{x}_2 &= v_2, \\
m_2 \dot{v}_2 &= -c_2 x_2 - d_2 v_2 - c_c(x_2 - x_1) - d_c(v_2 - v_1).
\end{aligned} \tag{11.6}$$

In Ref. [7], three different approaches for decomposing the overall system into two subsystems are described:

- Displacement/displacement-decomposition: Both subsystems are base-point excited single-mass oscillators. With the coupling function $\mathbf{u} = [x_1 \ v_1 \ x_2 \ v_2]^T$ and the coupling vector $\mathbf{p} = [\tilde{x}_1 \ \tilde{v}_1 \ \tilde{x}_2 \ \tilde{v}_2]^T$ the decomposed system is given by

Subsystem 1:

$$\begin{aligned}
\dot{x}_1 &= v_1, \\
m_1 \dot{v}_1 &= -c_1 x_1 - d_1 v_1 + c_c(\tilde{x}_2 - x_1) + d_c(\tilde{v}_2 - v_1),
\end{aligned} \tag{11.7}$$

Subsystem 2:

$$\begin{aligned}
\dot{x}_2 &= v_2, \\
m_2 \dot{v}_2 &= -c_2 x_2 - d_2 v_2 - c_c(x_2 - \tilde{x}_1) - d_c(v_2 - \tilde{v}_1),
\end{aligned} \tag{11.8}$$

Coupling Conditions:

$$\begin{aligned}
g_{x_1}^{\text{co}} &:= \tilde{x}_1 - x_1 = 0, \\
g_{v_1}^{\text{co}} &:= \tilde{v}_1 - v_1 = 0, \\
g_{x_2}^{\text{co}} &:= \tilde{x}_2 - x_2 = 0, \\
g_{v_2}^{\text{co}} &:= \tilde{v}_2 - v_2 = 0.
\end{aligned} \tag{11.9}$$

- Force/force-decomposition: An alternative approach is obtained by using the coupling force as coupling variable. Then, both subsystems are force-driven single-mass oscillators. With the one-dimensional coupling function $\mathbf{u} = [c_c(x_2 - x_1) + d_c(v_2 - v_1)]$ and the one-dimensional coupling vector $\mathbf{p} = [\lambda_c]$ the decomposed system is defined by

Subsystem 1:

$$\begin{aligned}
\dot{x}_1 &= v_1, \\
m_1 \dot{v}_1 &= -c_1 x_1 - d_1 v_1 + \lambda_c,
\end{aligned} \tag{11.10}$$

Subsystem 2:

$$\begin{aligned}\dot{x}_2 &= v_2, \\ m_2 \dot{v}_2 &= -c_2 x_2 - d_2 v_2 - \lambda_c,\end{aligned}\tag{11.11}$$

Coupling Condition:

$$g_\lambda^{\text{co}} := \lambda_c - c_c(x_2 - x_1) - d_c(v_2 - v_1) = 0.\tag{11.12}$$

- **Force/displacement-decomposition:** The third approach is a mixture of the two other approaches. The first subsystem is a force-driven single-mass oscillator and the second subsystem is a base-point excited single-mass oscillator. With the coupling function $\mathbf{u} = [x_1 \ v_1 \ c_c(x_2 - x_1) + d_c(v_2 - v_1)]^T$ and the coupling vector $\mathbf{p} = [\tilde{x}_1 \ \tilde{v}_1 \ \lambda_c]^T$ the decomposed system is given by

Subsystem 1:

$$\begin{aligned}\dot{x}_1 &= v_1, \\ m_1 \dot{v}_1 &= -c_1 x_1 - d_1 v_1 + \lambda_c,\end{aligned}\tag{11.13}$$

Subsystem 2:

$$\begin{aligned}\dot{x}_2 &= v_2, \\ m_2 \dot{v}_2 &= -c_2 x_2 - d_2 v_2 - c_c(x_2 - \tilde{x}_1) - d_c(v_2 - \tilde{v}_1),\end{aligned}\tag{11.14}$$

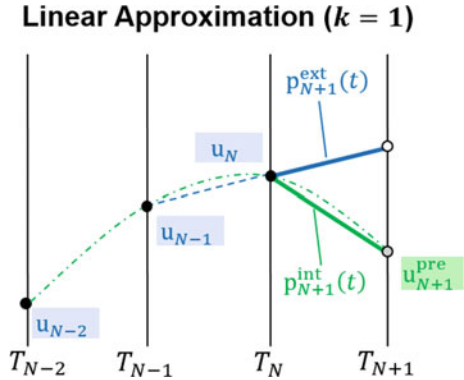
Coupling Conditions:

$$\begin{aligned}g_{x_1}^{\text{co}} &:= \tilde{x}_1 - x_1 = 0, \\ g_{v_1}^{\text{co}} &:= \tilde{v}_1 - v_1 = 0, \\ g_\lambda^{\text{co}} &:= \lambda_c - c_c(x_2 - x_1) - d_c(v_2 - v_1) = 0.\end{aligned}\tag{11.15}$$

11.4 Two Explicit Co-simulation Approaches

In this section, two different methods for approximating the coupling variables are explained. Both methods are illustrated in Fig. 11.2, where the value of the coupling function at the communication-time points are denoted by $\mathbf{u}_N = \mathbf{u}(T_N, \mathbf{z}_N)$. The degree of the approximation polynomials is denoted by k . The approximation polynomials for the integration from T_N to T_{N+1} are indicated by the subscript $N + 1$. To distinguish between the approximation polynomials of the two different methods, the approximation polynomials are extended by appropriate superscripts $(\cdot)^{\text{ext}}$ or $(\cdot)^{\text{int}}$.

Fig. 11.2 Approximation polynomials for method 1 and method 2



11.4.1 Method 1

The first approach makes use of a classical extrapolation technique, see Ref. [7]. The approximation polynomials are denoted by $\mathbf{p}_{N+1}^{ext}(t)$. Carrying out the co-simulation with polynomials of degree k , the vector $\mathbf{p}_{N+1}^{ext}(t)$ extrapolates $\mathbf{u}(t, \mathbf{z}(t))$ in the interval from T_N to T_{N+1} by the $k + 1$ supporting points $\mathbf{p}_{N+1}^{ext}(T_{N-k}) = \mathbf{u}_{N-k}, \dots, \mathbf{p}_{N+1}^{ext}(T_N) = \mathbf{u}_N$.

11.4.2 Method 2

Using the second method, the approximation polynomials are termed by $\mathbf{p}_{N+1}^{int}(t)$. At first, a predictor vector \mathbf{u}_{N+1}^{pre} is calculated, which extrapolates $\mathbf{u}(T_{N+1}, \mathbf{z}_{N+1})$ by the $k + 2$ (or more) sampling points $\mathbf{u}_{N-k-1}, \dots, \mathbf{u}_N$ with the help of the *Neville-Aitken* scheme. Then, the vector $\mathbf{p}_{N+1}^{int}(t)$ is constructed, which interpolates $\mathbf{u}(t, \mathbf{z}(t))$ from T_N to T_{N+1} by the $k + 1$ supporting points $\mathbf{p}_{N+1}^{int}(T_{N-k+1}) = \mathbf{u}_{N-k+1}, \dots, \mathbf{p}_{N+1}^{int}(T_N) = \mathbf{u}_N, \mathbf{p}_{N+1}^{int}(T_{N+1}) = \mathbf{u}_{N+1}^{pre}$.

11.4.3 Error Estimation Approach

The idea is to execute each subsystem integration twice: once with the polynomials $\mathbf{p}_{N+1}^{ext}(t)$ and secondly with the polynomials $\mathbf{p}_{N+1}^{int}(t)$. The results of the first simulation, are used as initial values for the next integration from T_{N+1} to T_{N+2} . The results of the second simulation only serve as reference for an error estimation. In each communication-time step, the initial conditions the reference are reinitialized with the corresponding values of the first integration. Thus, both integrations have the same initial conditions at the communication-time points. Since the integrations

are executed in parallel, there is only little extra computation time necessary. It can be shown that both methods have the same convergence order. Computing the error constants for both methods, the local error of the co-simulation is estimated with the help of the *Milne*-device approach [6].

11.5 Error Analysis of Co-simulation

For the error analysis of the co-simulation, it is assumed that the subsystems are solved exactly between two consecutive communication-time points T_N and T_{N+1} . Hence, the error analysis only deals with the error generated by the co-simulation, i.e. by the approximation of the coupling variables.

Since we make use of an applied-force coupling approach, all constraints of the overall system are inner constraints of the subsystems. For the following error analysis, it is necessary to transfer the subsystem DAEs into corresponding underlying ODEs. Since the subsystem integrations are assumed to be exact in our analysis, the error of the co-simulation is analyzed for ODE-subsystems, supposing that the equations of motion of the subsystems are independent of derivatives of the coupling variables.

Example 11.1 The equations of motion of a mechanical subsystem with scleronomic constraints are given by

$$\dot{\mathbf{q}}^s = \mathbf{K}^s(\mathbf{q}^s) \mathbf{v}^s, \quad (11.16a)$$

$$\mathbf{M}^s(t, \mathbf{q}^s) \dot{\mathbf{v}}^s = \mathbf{f}^{e,s}(t, \mathbf{q}^s, \mathbf{v}^s, \mathbf{p}(t)) - \Phi^s(\mathbf{q}^s)^T \boldsymbol{\lambda}^s, \quad (11.16b)$$

$$\mathbf{0} = \phi^s(\mathbf{q}^s). \quad (11.16c)$$

with $\Phi^s(\mathbf{q}^s) := \phi_{\mathbf{q}^s}^s(\mathbf{q}^s) \mathbf{K}^s(\mathbf{q}^s)$. Differentiating Eq. (11.16c) twice, we get together with Eq. (11.16b) the index-1 system

$$\begin{bmatrix} \mathbf{M}^s(t, \mathbf{q}^s) & \Phi^s(\mathbf{q}^s)^T \\ \Phi^s(\mathbf{q}^s) & \mathbf{0} \end{bmatrix} \begin{bmatrix} \dot{\mathbf{v}}^s \\ \boldsymbol{\lambda}^s \end{bmatrix} = \begin{bmatrix} \mathbf{f}^{e,s}(t, \mathbf{q}^s, \mathbf{v}^s, \mathbf{p}(t)) \\ -(\Phi^s(\mathbf{q}^s) \mathbf{v}^s)_{\mathbf{q}^s} \dot{\mathbf{q}}^s \end{bmatrix}. \quad (11.17)$$

If the matrix on the left hand side of Eq.(11.17) is non-singular, Eqs.(11.16a), (11.16b), and (11.16c) can be transformed into an ODE-subsystem of the form

$$\dot{\mathbf{y}}^s = \mathbf{F}^{\text{co},s}(t, \mathbf{y}^s, \mathbf{p}(t)), \quad (11.18)$$

where \mathbf{y}^s contains the position variables \mathbf{q}^s and the velocity variables \mathbf{v}^s of subsystem s . The *Lagrange* multipliers are eliminated.

11.5.1 Consistency and Zero Stability of the Co-simulation

The consistency order describes the perturbation of the analytical solution by the numerical procedure. Neglecting the errors of the subsystem solvers, the consistency error is generated by the approximation of the coupling variables. For a consistency analysis, the integration from T_N to T_{N+1} is carried out riding on the assumption that the analytical solutions of the state variables are known. The coupling variables are substituted by approximation polynomials as described in Sect. 11.4.

Theorem 11.1 (Consistency) *We consider the initial value problem with the equations of motion*

$$\begin{aligned}\dot{\mathbf{y}} &= \mathbf{F}^{\text{co}}(t, \mathbf{y}, \mathbf{p}), \\ \mathbf{0} &= \mathbf{g}^{\text{co}}(t, \mathbf{y}, \mathbf{p}),\end{aligned}\tag{11.19}$$

where $\mathbf{g}^{\text{co}}(t, \mathbf{y}, \mathbf{p}) = \mathbf{p} - \mathbf{u}(t, \mathbf{y})$ defines the right-hand side of the coupling conditions and $\mathbf{y}(t_0) = \mathbf{y}_0$ the initial conditions. It is assumed that $\mathbf{F}^{\text{co}}(t, \mathbf{y}, \mathbf{u}(t, \mathbf{y}))$ is smooth enough and globally Lipschitz-continuous with respect to \mathbf{y} . Let $T_0 = t_0, \dots, T_N = t_0 + H \cdot N, \dots$ be the equidistant communication-time grid with communication-step size H and let the values of the coupling function $\mathbf{u}_{N-k-1} = \mathbf{u}(T_{N-k-1}, \mathbf{y}_{N-k-1}), \dots, \mathbf{u}_{N+1} = \mathbf{u}(T_{N+1}, \mathbf{y}_{N+1})$ be computed by exact values $\mathbf{y}_{N-k-1} = \mathbf{y}(T_{N-k-1}), \dots, \mathbf{y}_{N+1} = \mathbf{y}(T_{N+1})$. Then, both approaches explained in Sect. 11.4 are consistent with order $k + 1$.

Proof The polynomials $\mathbf{p}_{N+1}^{\text{ext}}(t)$ (method 1) approximate the coupling variables with an error

$$\|\mathbf{p}_{N+1}^{\text{ext}}(t) - \mathbf{p}(t)\| = \mathcal{O}(H^{k+1}).\tag{11.20}$$

The predictor value $\mathbf{u}_{N+1}^{\text{pre}}$ (method 2) is an approximation of the coupling variables at the communication-time point T_{N+1} with an error

$$\|\mathbf{u}_{N+1}^{\text{pre}} - \mathbf{u}_{N+1}\| = \mathcal{O}(H^{k+2}).\tag{11.21}$$

Consequently, the approximation of the coupling variables with the polynomials $\mathbf{p}_{N+1}^{\text{int}}(t)$ yields

$$\|\mathbf{p}_{N+1}^{\text{int}}(t) - \mathbf{p}(t)\| = \mathcal{O}(H^{k+1}).\tag{11.22}$$

Equations (11.20) and (11.22) imply

$$\|\mathbf{F}^{\text{co}}(t, \mathbf{y}(t), \mathbf{p}_{N+1}^*(t)) - \dot{\mathbf{y}}(t)\| = \mathcal{O}(H^{k+1}),\tag{11.23}$$

where $(\cdot)^*$ represents one of the superscripts $(\cdot)^{\text{ext}}$ or $(\cdot)^{\text{int}}$. Generally, we have

$$\mathbf{y}_{N+1} = \mathbf{y}_N + \int_{T_N}^{T_{N+1}} \dot{\mathbf{y}}(t) dt .$$

Accordingly, we define

$$\mathbf{y}_{N+1}^* := \mathbf{y}_N + \int_{T_N}^{T_{N+1}} \mathbf{F}^{\text{co}}(t, \mathbf{y}(t), \mathbf{p}_{N+1}^*(t)) dt .$$

Therefore, from Eq.(11.23) we can deduce that

$$\begin{aligned} \frac{1}{H} \|\mathbf{y}_{N+1}^* - \mathbf{y}_{N+1}\| &= \frac{1}{H} \left\| \int_{T_N}^{T_{N+1}} \mathbf{F}^{\text{co}}(t, \mathbf{y}(t), \mathbf{p}_{N+1}^*(t)) - \dot{\mathbf{y}}(t) dt \right\| \\ &\leq \frac{1}{H} \int_{T_N}^{T_{N+1}} \|\mathbf{F}^{\text{co}}(t, \mathbf{y}(t), \mathbf{p}_{N+1}^*(t)) - \dot{\mathbf{y}}(t)\| dt \\ &= \mathcal{O}(H^{k+1}) . \end{aligned} \quad (11.24)$$

Hence, we obtain

$$\begin{aligned} \frac{1}{H} \|\mathbf{u}(T_{N+1}, \mathbf{y}_{N+1}^*) - \mathbf{u}_{N+1}\| &= \frac{1}{H} \|\mathbf{u}(T_{N+1}, \mathbf{y}_{N+1}^*) - \mathbf{u}(T_{N+1}, \mathbf{y}_{N+1})\| \\ &= \mathcal{O}(H^{k+1}) . \end{aligned} \quad (11.25)$$

Finally, Eqs.(11.24) and (11.25) imply that both methods are consistent with order $k + 1$. \square

Zero stability follows from the fact that the state variables at the previous communication-time points (T_{N-1}, T_{N-2}, \dots) will only effect the coupling variables, but not the state variables [4]. The subsystem integrations are independent of the coupling variables for vanishing communication-step size H , since

$$\mathbf{y}(T_{N+1}) = \mathbf{y}(T_N) + H\mathbf{F}^{\text{co}}(T_N, \mathbf{y}(T_N), \mathbf{p}_{N+1}^*(T_N)) + \mathcal{O}(H^2) . \quad (11.26)$$

Therefore, the error of both methods converges with order $k + 1$.

Figure 11.3 shows convergence plots for the 2-DOF oscillator of Fig. 11.1 based on a force/force-decomposition technique. The plots verify that both methods converge with order $k + 1$. The convergence plots in the first row, show the global error of the position variables for various communication-step sizes. The global error of the velocity variables is depicted in the lower convergence plots. The left column refers to method 1, the right column corresponds to method 2. The simulations have been carried out with an error tolerance of 10^{-13} for the subsystem integrations,

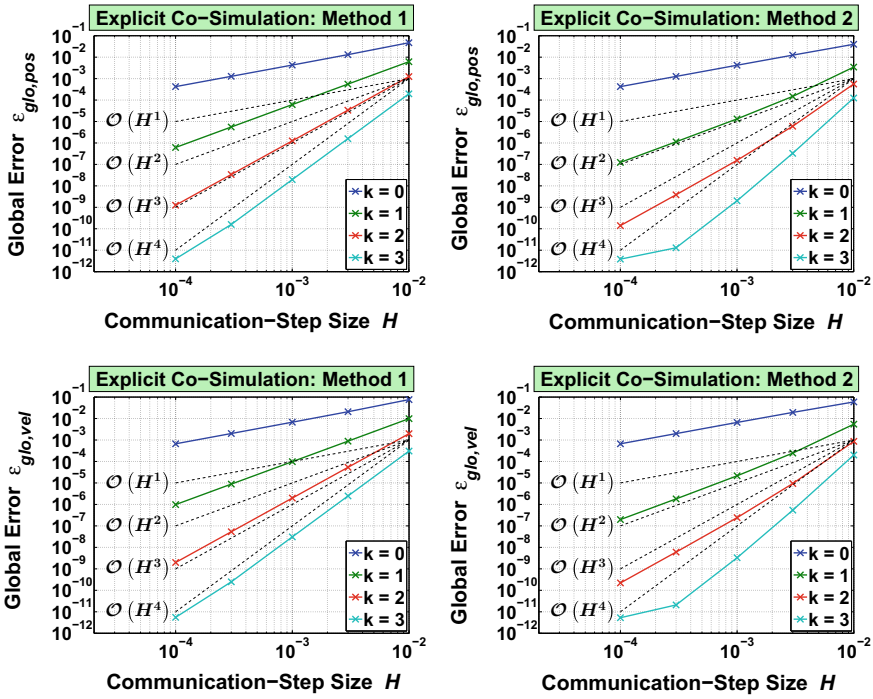


Fig. 11.3 Convergence of the global error for the two co-simulation methods presented in Sect. 11.4 (position and velocity level)

which causes a kink in the convergence plot of method 2 carried out with cubic approximation polynomials ($k = 3$). The subsystems are solved with a third-fourth order *Rosenbrock–Wanner* method.

11.5.2 Local Error of the Co-simulation

Now, the local error of the state variables between the communication-time points T_N and T_{N+1} is investigated. Let $\mathbf{y}^*(t)$ denote the solution carried out with the polynomial $\mathbf{p}_{N+1}^*(t)$ and $\mathbf{y}(t)$ the exact solution of the trajectory with $\mathbf{y}(T_N) = \mathbf{y}_N$. Since both methods presented in Sect. 11.4 are consistent with order $k + 1$ and zero-stable, the numerical solutions $\mathbf{y}_{N-k-1}, \dots, \mathbf{y}_{N-1}$ agree with the exact solution with order $\mathcal{O}(H^{k+2})$ at the communication-time points $T_{N-k-1}, \dots, T_{N-1}$, where $H = T_{N+1} - T_N$ denotes the current communication-step size supposing that $T_{N+1} - T_{N-k-1} = \mathcal{O}(H)$. Hence, the polynomials $\mathbf{p}_{N+1}^*(t)$ approximate the coupling variables, as well as the corresponding derivatives, with an error

$$\left\| \mathbf{p}_{N+1}^{*(j)}(t) - \mathbf{p}^{(j)}(t) \right\| = \mathcal{O}(H^{k+1-j}) \quad (11.27)$$

for $j = 0, \dots, k$, where $(\cdot)^{(j)}$ denotes the j th time derivative. According to Eq. (11.19.1) $\dot{\mathbf{y}}$ depends on \mathbf{p} , but $\dot{\mathbf{y}}$ is independent of $\dot{\mathbf{p}}$ and higher order derivatives. By induction, we can show that $\mathbf{y}^{(j)}$ depends on $\mathbf{p}^{(j-1)}$, but $\mathbf{y}^{(j)}$ is independent of $\mathbf{p}^{(j)}$ and higher order derivatives. The base case ($j = 1$) is given by Eq. (11.19). By induction hypothesis, $\mathbf{y}^{(j)}$ can be written as a function according to

$$\mathbf{y}^{(j)} = \mathcal{F}^j(t, \mathbf{y}; \mathbf{p}, \dots, \mathbf{p}^{(j-1)}). \quad (11.28)$$

Differentiating Eq. (11.28) yields

$$\begin{aligned} \mathbf{y}^{(j+1)} = & \mathcal{F}_t^j(t, \mathbf{y}; \mathbf{p}, \dots, \mathbf{p}^{(j-1)}) + \mathcal{F}_y^j(t, \mathbf{y}; \mathbf{p}, \dots, \mathbf{p}^{(j-1)}) \dot{\mathbf{y}} \\ & + \sum_{i=0}^{j-1} \mathcal{F}_{\mathbf{p}^{(i)}}^j(t, \mathbf{y}; \mathbf{p}, \dots, \mathbf{p}^{(j-1)}) \mathbf{p}^{(i+1)}, \end{aligned} \quad (11.29)$$

which implies that the statement is true. Together with Eq. (11.27), we have

$$\left\| \mathbf{y}^{*(j+1)}(T_N) - \mathbf{y}^{(j+1)}(T_N) \right\| = \mathcal{O}(H^{k+1-j}). \quad (11.30)$$

Finally, a *Taylor*-series expansion provides

$$\begin{aligned} \left\| \mathbf{y}^*(t) - \mathbf{y}(t) \right\| & \leq \sum_{j=0}^k \frac{|t-T_N|^j}{j!} \left\| \mathbf{y}^{*(j)}(T_N) - \mathbf{y}^{(j)}(T_N) \right\| + \mathcal{O}(H^{k+2}) \\ & = \mathcal{O}(H^{k+2}) \end{aligned} \quad (11.31)$$

for $t \in (T_N, T_{N+1})$.

Figure 11.4 illustrates the convergence of the local errors. It can be seen that the local errors of the velocity variables converge with order $k + 2$ for both methods. The local errors of the position variables even converge with order $k + 3$. Note that a higher convergence order of the local error is achieved, since \dot{x}_1 and \dot{x}_2 in Eqs. (11.10) and (11.11) are independent of the coupling variables.

11.6 Coupling of Different Multibody Systems with an Applied-Force Coupling Approach

Assuming that each subsystem is described by a constrained mechanical system, the equations of motion of subsystem $s = 1, \dots, r$ read as

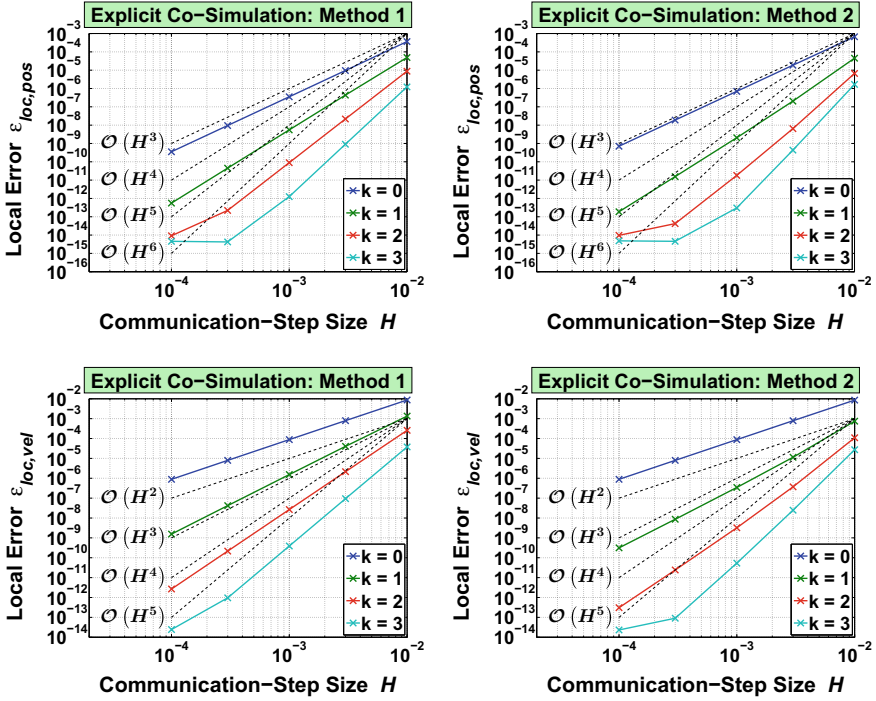


Fig. 11.4 Convergence of the local error of the two co-simulation methods presented in Sect. 11.4

$$\begin{aligned}
 \dot{\mathbf{q}}^s &= \mathbf{K}^s(t, \mathbf{q}^s) \mathbf{v}^s, \\
 \mathbf{M}^s(t, \mathbf{q}^s) \dot{\mathbf{v}}^s &= \mathbf{f}^{\text{co},s}(t, \mathbf{q}^s, \mathbf{v}^s, \boldsymbol{\lambda}^s, \mathbf{p}(t)), \\
 \mathbf{0} &= \boldsymbol{\phi}^s(t, \mathbf{q}^s, \mathbf{v}^s, \boldsymbol{\lambda}^s),
 \end{aligned} \tag{11.32}$$

where $\mathbf{p}(t)$ denotes a vector of polynomials approximating $\mathbf{u}(t, \mathbf{q}(t), \mathbf{v}(t))$ between two consecutive communication-time points. Since an applied-force coupling approach is considered, the coupling variables are assumed to be independent of the *Lagrange* multipliers $\boldsymbol{\lambda}^s$ for $s = 1, \dots, r$. At the communication-time point T_N , the subsystems are coupled by the coupling condition

$$\mathbf{g}^{\text{co}}(T_N, \mathbf{q}_N, \mathbf{v}_N, \mathbf{p}_N) := \mathbf{p}_N - \mathbf{u}(T_N, \mathbf{q}_N, \mathbf{v}_N) = \mathbf{0} \tag{11.33}$$

with $\mathbf{q}_N = \mathbf{q}(T_N)$, $\mathbf{v}_N = \mathbf{v}(T_N)$, and $\mathbf{p}_N = \mathbf{p}(T_N)$. The matrix $\mathbf{K}^s(t, \mathbf{q}^s)$ describes the relationship between the velocity variables and the derivatives of the position variables. The matrix $\mathbf{M}^s(t, \mathbf{q}^s)$ denotes the mass matrix of subsystem s . The inner (not necessarily holonomic) constraints of subsystem s are denoted by the vector $\boldsymbol{\phi}^s(t, \mathbf{q}^s, \mathbf{v}^s, \boldsymbol{\lambda}^s)$. The vector $\mathbf{f}^s(t, \mathbf{q}^s, \mathbf{v}^s, \boldsymbol{\lambda}^s, \mathbf{p}(t))$ contains the forces and torques in subsystem s . According to Eq. (11.18), we assume that the accelerations and the

Lagrange multipliers can be interpreted as functions, which depend amongst others on the coupling variables, but are independent of derivatives of the coupling variables, that is

$$\begin{aligned}\ddot{\mathbf{q}}^s &= \mathbf{a}^s(t, \mathbf{q}^s, \mathbf{v}^s, \mathbf{p}(t)), \\ \dot{\mathbf{v}}^s &= \mathbf{b}^s(t, \mathbf{q}^s, \mathbf{v}^s, \mathbf{p}(t)), \\ \boldsymbol{\lambda}^s &= \boldsymbol{\Lambda}^s(t, \mathbf{q}^s, \mathbf{v}^s, \mathbf{p}(t)).\end{aligned}\tag{11.34}$$

The equations of motion of the decomposed overall system read as

$$\begin{aligned}\dot{\mathbf{q}} &= \mathbf{K}(t, \mathbf{q}) \mathbf{v}, \\ \mathbf{M}(t, \mathbf{q}) \dot{\mathbf{v}} &= \mathbf{f}^{\text{co}}(t, \mathbf{q}, \mathbf{v}, \boldsymbol{\lambda}, \mathbf{p}(t)), \\ \mathbf{0} &= \boldsymbol{\phi}(t, \mathbf{q}, \mathbf{v}, \boldsymbol{\lambda})\end{aligned}\tag{11.35}$$

with the coupling condition (11.33) at the communication-time points. The accelerations and the *Lagrange* multipliers of all subsystems are collected in the vectors

$$\begin{aligned}\ddot{\mathbf{q}} &= \mathbf{a}(t, \mathbf{q}, \mathbf{v}, \mathbf{p}(t)), \\ \dot{\mathbf{v}} &= \mathbf{b}(t, \mathbf{q}, \mathbf{v}, \mathbf{p}(t)), \\ \boldsymbol{\lambda} &= \boldsymbol{\Lambda}(t, \mathbf{q}, \mathbf{v}, \mathbf{p}(t)).\end{aligned}\tag{11.36}$$

Then, the underlying ODE reads as

$$\begin{bmatrix} \dot{\mathbf{q}} \\ \dot{\mathbf{v}} \end{bmatrix} = \begin{bmatrix} \mathbf{K}(t, \mathbf{q}) \mathbf{v} \\ \mathbf{b}(t, \mathbf{q}, \mathbf{v}, \mathbf{p}(t)) \end{bmatrix}.\tag{11.37}$$

Under the assumption that the subsystem integrations are exact, approximating the coupling variables in the DAE-system (11.35), and in the ODE-system (11.37), integrations in the interval (T_N, T_{N+1}) will yield the same solutions. Hence, the error analysis in Sect. 11.5 is valid for the DAE-system (11.35), if assumption (11.34) holds in each subsystem.

11.7 Co-simulation Approach with Variable Communication-Step Size

In this section, error constants of the two methods presented in Sect. 11.4 are derived in order to construct an error estimator with the help of the *Milne*-device approach.

11.7.1 Local Truncation Error

The numerical solutions at the communication-time point T_N for all subsystems are collected in the vectors $\mathbf{q}_N, \mathbf{v}_N, \mathbf{\Lambda}_N$. We consider the integration from T_N to T_{N+1} . Let $\mathbf{q}^*(t), \mathbf{v}^*(t), \mathbf{\Lambda}^*(t)$ describe the solution carried out with the approximation polynomial $\mathbf{p}_{N+1}^*(t)$ and let $\mathbf{q}(t), \mathbf{v}(t), \mathbf{\Lambda}(t), \mathbf{p}(t)$ denote the exact solutions with respect to the initial conditions $\mathbf{q}(T_N) = \mathbf{q}_N$ and $\mathbf{v}(T_N) = \mathbf{v}_N$. Note that $\mathbf{p}(t) = \mathbf{u}(t, \mathbf{q}(t), \mathbf{v}(t))$ is not necessarily polynomial. Using a *Taylor-series* expansion and Eq. (11.31), we obtain

$$\begin{aligned} \mathbf{q}_{N+1}^* &:= \mathbf{q}^*(T_{N+1}) = \mathbf{q}_N + H \cdot \mathbf{K}(T_N, \mathbf{q}_N) \mathbf{v}_N \\ &\quad + \int_{T_N}^{T_{N+1}} \int_{T_N}^{\tau} \mathbf{a}(t, \mathbf{q}^*(t), \mathbf{v}^*(t), \mathbf{p}_{N+1}^*(t)) dt d\tau \\ &= \mathbf{q}_N + H \cdot \mathbf{K}(T_N, \mathbf{q}_N) \mathbf{v}_N \\ &\quad + \int_{T_N}^{T_{N+1}} \int_{T_N}^{\tau} \mathbf{a}(t, \mathbf{q}(t), \mathbf{v}(t), \mathbf{p}_{N+1}^*(t)) dt d\tau + \mathcal{O}(H^{k+4}). \end{aligned} \quad (11.38)$$

Equation (11.27) with $j = 0$ yields

$$\begin{aligned} &\mathbf{q}_{N+1}^* - \mathbf{q}(T_{N+1}) \\ &= \int_{T_N}^{T_{N+1}} \int_{T_N}^{\tau} \mathbf{a}_{\mathbf{p}}(t, \mathbf{q}(t), \mathbf{v}(t), \mathbf{p}(t)) (\mathbf{p}_{N+1}^*(t) - \mathbf{p}(t)) dt d\tau + \mathcal{O}(H^{k+4}) \\ &= \mathbf{a}_{\mathbf{p}}(T_N, \mathbf{q}_N, \mathbf{v}_N, \mathbf{u}_N) \int_{T_N}^{T_{N+1}} \int_{T_N}^{\tau} \mathbf{p}_{N+1}^*(t) - \mathbf{p}(t) dt d\tau + \mathcal{O}(H^{k+4}) \\ &= \mathbf{a}_{\mathbf{p},N} \int_{T_N}^{T_{N+1}} \int_{T_N}^{\tau} \mathbf{p}_{N+1}^*(t) - \mathbf{p}(t) dt d\tau + \mathcal{O}(H^{k+4}), \end{aligned} \quad (11.39)$$

where $\mathbf{a}_{\mathbf{p},N} := \mathbf{a}_{\mathbf{p}}(T_N, \mathbf{q}_N, \mathbf{v}_N, \mathbf{u}_N)$ denotes the *Jacobian*-matrix of the accelerations with respect to the coupling variables at the communication-time point T_N . For computing the error constants, polynomials $\hat{\mathbf{p}}_{N+1}(t)$ of degree $k + 1$ are introduced, extrapolating the coupling variables by the $k + 2$ supporting points $\hat{\mathbf{p}}_{N+1}(T_{N-k-1}) = \mathbf{u}_{N-k-1}, \dots, \hat{\mathbf{p}}_{N+1}(T_N) = \mathbf{u}_N$. Further, $L_{N+1}^i(t)$ denote the *Lagrange*-basis polynomials

$$L_{N+1}^i(t) = \prod_{j=0}^{i-1} \frac{t - T_{N-j}}{T_{N+1} - T_{N-j}} \quad (i \in \{k, k+1\}). \quad (11.40)$$

Note that $\hat{\mathbf{p}}_{N+1}(T_{N+1})$ equal the predictor values $\mathbf{u}_{N+1}^{\text{pre}} = \mathbf{p}_{N+1}^{\text{int}}(T_{N+1})$. Since $\hat{\mathbf{p}}_{N+1}(t)$ and $\mathbf{p}_{N+1}^{\text{ext}}(t)$ intersect at the communication-time points T_{N-k}, \dots, T_N and since $\mathbf{p}_{N+1}^{\text{int}}(t)$ and $\mathbf{p}_{N+1}^{\text{ext}}(t)$ intersect at T_{N-k+1}, \dots, T_N , we have

$$\mathbf{p}_{N+1}^{\text{ext}}(t) - \hat{\mathbf{p}}_{N+1}(t) = \Delta \mathbf{p}_{N+1} L_{N+1}^{k+1}(t), \quad (11.41)$$

$$\mathbf{p}_{N+1}^{\text{ext}}(t) - \mathbf{p}_{N+1}^{\text{int}}(t) = \Delta \mathbf{p}_{N+1} L_{N+1}^k(t) \quad (11.42)$$

for $\Delta \mathbf{p}_{N+1} = \mathbf{p}_{N+1}^{\text{ext}}(T_{N+1}) - \mathbf{p}_{N+1}^{\text{int}}(T_{N+1})$. Since both methods presented in Sect. 11.4 are consistent with order $k+1$ and zero-stable, the numerical solutions $\mathbf{q}_{N-k-1}, \dots, \mathbf{q}_{N-1}$ and $\mathbf{v}_{N-k-1}, \dots, \mathbf{v}_{N-1}$ agree with the exact solution with order $\mathcal{O}(H^{k+2})$ at the communication-time points $T_{N-k-1}, \dots, T_{N-1}$. Hence, the polynomials $\hat{\mathbf{p}}_{N+1}(t)$ approximate the coupling variables with an error

$$\|\hat{\mathbf{p}}_{N+1}(t) - \mathbf{p}(t)\| = \mathcal{O}(H^{k+2}). \quad (11.43)$$

Therefore, Eq. (11.39) can be rewritten as

$$\mathbf{q}_{N+1}^* - \mathbf{q}(T_{N+1}) = \mathbf{a}_{\mathbf{p},N} \int_{T_N}^{T_{N+1}} \int_{T_N}^{\tau} \mathbf{p}_{N+1}^*(t) - \hat{\mathbf{p}}_{N+1}(t) dt d\tau + \mathcal{O}(H^{k+4}). \quad (11.44)$$

Using Eqs. (11.41) and (11.42), the local errors are given by

$$\mathbf{q}_{N+1}^{\text{ext}} - \mathbf{q}(T_{N+1}) = C_{N+1}^{k+1} \mathbf{a}_{\mathbf{p},N} \Delta \mathbf{p}_{N+1} + \mathcal{O}(H^{k+4}), \quad (11.45)$$

$$\mathbf{q}_{N+1}^{\text{int}} - \mathbf{q}(T_{N+1}) = (C_{N+1}^{k+1} - C_{N+1}^k) \mathbf{a}_{\mathbf{p},N} \Delta \mathbf{p}_{N+1} + \mathcal{O}(H^{k+4}) \quad (11.46)$$

with the constants

$$C_{N+1}^i := \int_{T_N}^{T_{N+1}} \int_{T_N}^{\tau} L_{N+1}^i(t) dt d\tau \quad (i \in \{k, k+1\}) \quad (11.47)$$

obtained by integrating the *Lagrange*-basis polynomials (11.40) twice. Note that the error constants depend on the previous communication-step size ratios.

11.7.2 Error Estimation for the Co-simulation

With the help of the *Milne*-device approach, an error estimator is constructed. Subtracting Eq. (11.46) from Eq. (11.45) yields

$$\mathbf{q}_{N+1}^{\text{ext}} - \mathbf{q}_{N+1}^{\text{int}} = C_{N+1}^k \mathbf{a}_{\mathbf{p},N} \Delta \mathbf{p}_{N+1} + \mathcal{O}(H^{k+4}). \quad (11.48)$$

The local error of the co-simulation approach presented in Sect. 11.4.1 can therefore be estimated by

$$\hat{\epsilon}_{N+1}^{\text{ext}} := \frac{C_{N+1}^{k+1}}{C_{N+1}^k} \|\mathbf{q}_{N+1}^{\text{ext}} - \mathbf{q}_{N+1}^{\text{int}}\|. \tag{11.49}$$

Remark 11.1 (Error Estimation on Velocity Level) An error estimator for the velocity variables can be constructed in a simultaneous way. The error constants on velocity level (corresponding to Eq. (11.47)) are obtained by one single integration of the Lagrange basis polynomials (11.40).

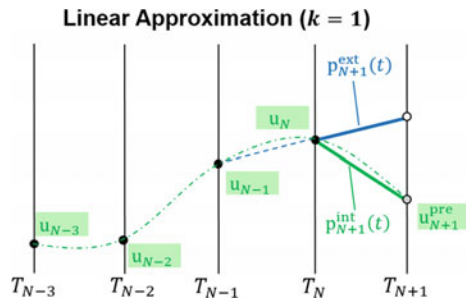
Remark 11.2 (Error Estimation with Role Reversal) In an alternative approach, the results of method 2 are used to continue the integration and the results of method 1 serve as reference for the error estimation. At the communication-time points, the integration of method 1 is reinitialized with the results of method 2 to get the same initial conditions for both methods. This yields the error estimator

$$\hat{\epsilon}_{N+1}^{\text{int}} := \left(1 - \frac{C_{N+1}^{k+1}}{C_{N+1}^k}\right) \|\mathbf{q}_{N+1}^{\text{ext}} - \mathbf{q}_{N+1}^{\text{int}}\|. \tag{11.50}$$

In our numerical tests, the error estimator of method 2 based on Eq. (11.50) works well for constant approximation polynomials ($k = 0$). For $k \geq 1$, error estimation of method 1 based on Eq. (11.49) showed better results than the estimation of method 2 based on Eq. (11.50). The error estimator is improved, if the extrapolation order for calculating $\mathbf{u}_{N+1}^{\text{pre}}$ is increased. If linear approximation polynomials ($k = 1$) are used for instance, the predicted values $\mathbf{u}_{N+1}^{\text{pre}}$ are approximated by at least three sampling points according to the description in Sect. 11.4.2. If the error estimator bases on Eq. (11.50) is used, it is recommended to approximate $\mathbf{u}_{N+1}^{\text{pre}}$ with at least four supporting points as illustrated in Fig. 11.5.

Remark 11.3 (Universal Error Constant) For reducing computation time, the user may calculate (inexact) universal constants for the case of an equidistant communication-step size, i.e.

Fig. 11.5 Approximation polynomials for method 1 and method 2 with higher extrapolation order for the predicted values of the coupling function



$$C^i := \frac{1}{i!} \int_0^1 \int_0^\tau \prod_{j=0}^{i-1} (t+j) dt d\tau \quad (i \in \{k, k+1\}). \quad (11.51)$$

Then,

$$\hat{\varepsilon}_{N+1}^{\text{ext}} := \frac{C^{k+1}}{C^k} \|\mathbf{q}_{N+1}^{\text{ext}} - \mathbf{q}_{N+1}^{\text{int}}\| \quad (11.52)$$

is used for error estimation in each communication-time step.

11.7.3 Using a Higher-Order Reference (Local Extrapolation)

Alternatively, an error estimator based on local extrapolation can be used [1]. In contrast to the algorithm described in Sect. 11.4, the reference may be carried out with the higher order polynomials $\hat{\mathbf{p}}_{N+1}(t)$. Hence, the reference solution is generated with approximation polynomials of degree $k+1$, which is extrapolated by $k+2$ sampling points. Then, the local truncation error is estimated by

$$\hat{\varepsilon}_{N+1}^{\text{ext}} := \|\mathbf{q}_{N+1}^{\text{ext}} - \hat{\mathbf{q}}_{N+1}\|, \quad (11.53)$$

where $\hat{\mathbf{q}}_{N+1}$ denotes the numerical solution, based on the approximation polynomials $\hat{\mathbf{p}}_{N+1}(t)$.

11.8 Numerical Examples

The error estimator is tested with the help of a 2-DOF oscillator (see Fig. 11.1). The simulations have been carried out with quadratic approximation polynomials ($k=2$). For the error estimation the error constants of Eq. (11.47) are used together with Eq. (11.49). Also, the universal error constants according to Eq. (11.51) are applied together with Eq. (11.52). Further, the error estimation based on the *Milne*-device approach (Sect. 11.7.2) is compared to the local extrapolation method described in Sect. 11.7.3. For controlling the communication-step size, a classical PI-step size controller is used [3].

11.8.1 Linear 2-DOF Oscillator

The first numerical example is the co-simulation test model presented in Sect. 11.3 based on a force/force-decomposition technique. The coupling variables are approxi-

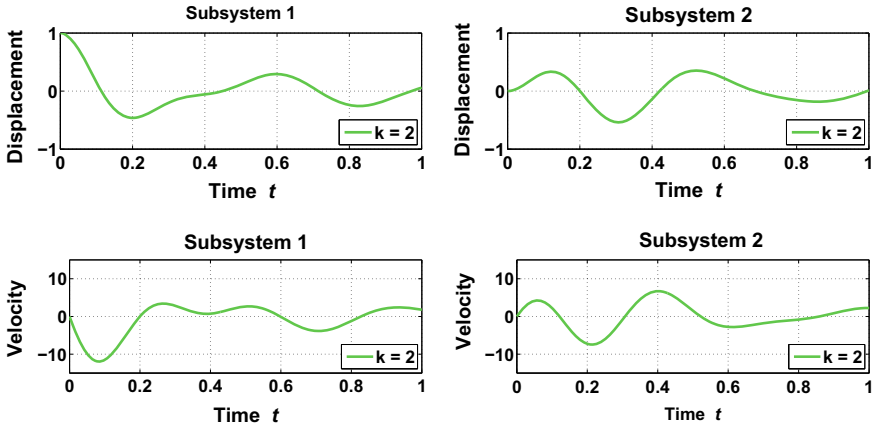


Fig. 11.6 Displacement and velocity of mass 1 and mass 2 of the linear two-mass oscillator

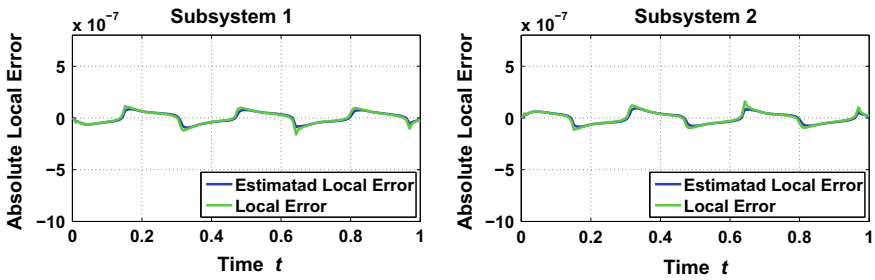


Fig. 11.7 Comparison of the estimated local error to the exact local error

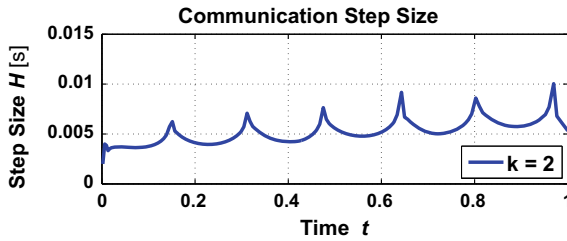


Fig. 11.8 Communication-step size of co-simulation of the linear 2-mass oscillator

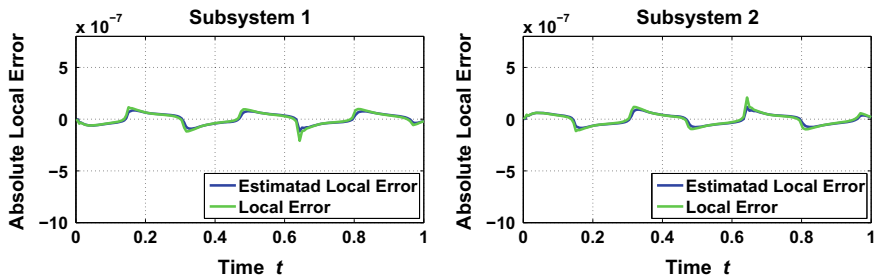
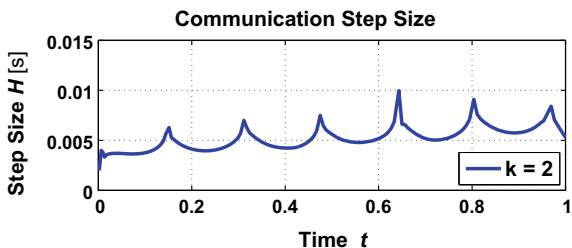


Fig. 11.9 Comparison of the estimated local error to the real local error (universal error constant)

Fig. 11.10 Communication-step size of co-simulation of the linear 2-mass oscillator using the universal error constant



mated by quadratic polynomials ($k = 2$). Since the 2-mass oscillator is a linear model, the local error can be computed analytically. Figure 11.6 illustrates the motions of both masses. The left column refers to mass 1, the right column corresponds to mass 2. The upper row shows the displacements of both masses. The lower plots depict the velocities. In Fig. 11.7, the estimated local error (blue line) is compared to the real local error (green line). It can be seen that the error estimator works well. The two lines are close together. Figure 11.8 illustrates the communication-step sizes of the simulation.

As mentioned in Remark 11.3 in Sect. 11.7.2, a universal error constant can be computed with the help of Eq. (11.51) in combination with Eq. (11.52). This simplified version is investigated in Figs. 11.9 and 11.10. In Fig. 11.9, the estimated local error (blue line) is compared to the real local error (green line). It can be seen that the error estimator with a universal error constant works almost as well as the error estimator with the real error constant according to Eq. (11.47) in combination with Eq. (11.49). Figure 11.10 illustrates the communication-step sizes of the simulation.

Now, the error estimator according to Eq. (11.53) is investigated. Therefore, the reference solution is generated with third-order polynomials. Figure 11.11 shows the estimated local error and the real local error for this approach. The error estimator works also quite well. Figure 11.12 depicts the communication-step size of the simulation.

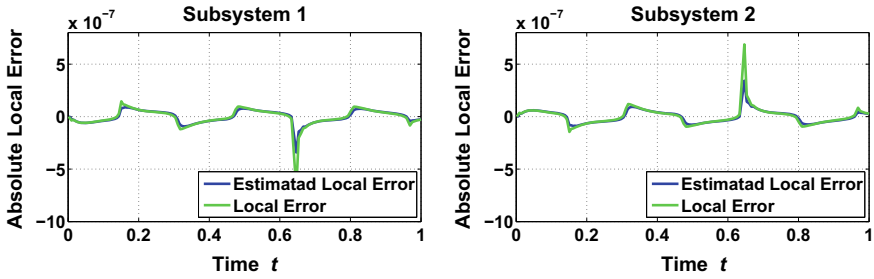


Fig. 11.11 Comparison of the estimated local error to the real local error of the co-simulation carried out with quadratic approximation polynomials; reference solution is carried out with cubic polynomials (see Sect. 11.7.3)

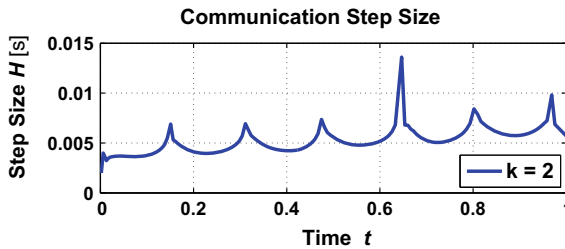


Fig. 11.12 Communication-step size of co-simulation carried out with quadratic approximation polynomials (reference carried out with cubic polynomials)

11.8.2 Non-linear 2-DOF Oscillator

To investigate the error estimation for a non-linear model, the motion of mass 1 is influenced by a contact at $x_1 = -0.1$. The contact is modeled by a penalty approach approximating the contact force with

$$F_{\text{ext}} = \exp(-10^6 \cdot (x_1(t) + 0.1)). \tag{11.54}$$

The error tolerances of the subsystem integrations are increased to 10^{-9} . Figure 11.13 illustrates the motions of both masses. Figure 11.14 illustrates the communication-step sizes of the simulation. As expected, the communication-step size is decreased at the contact points.

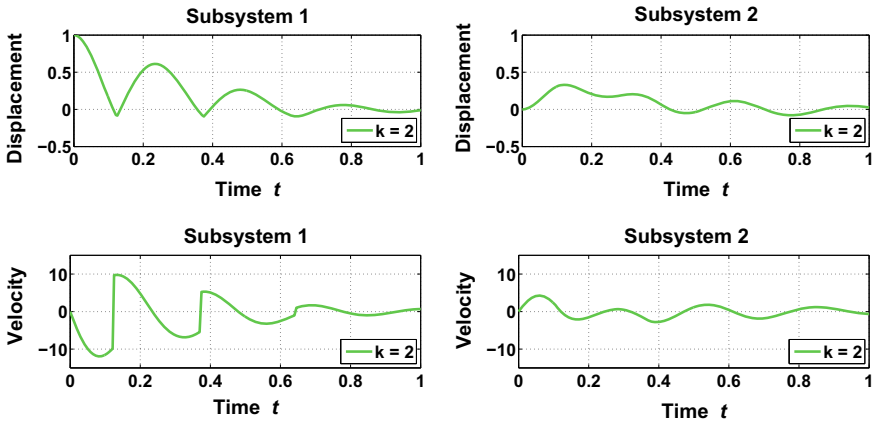
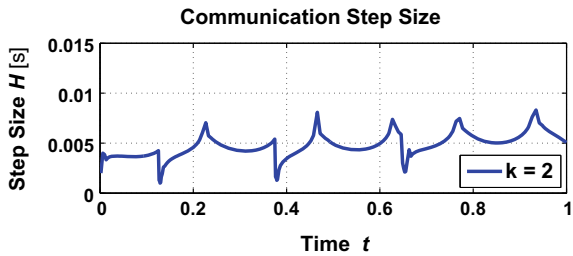


Fig. 11.13 Displacement and velocity of mass 1 and mass 2 for the non-linear two-mass oscillator

Fig. 11.14 Communication-step size of the co-simulation for the non-linear 2-mass oscillator



11.9 Conclusions

To generate an error estimator for explicit co-simulation approaches, each subsystem integration is carried out with two different methods of the same convergence order. Computing the error constants, an error estimator for controlling the communication-step size is constructed with the help of the *Milne*-device approach.

Alternatively, the subsystem integrations can be carried out with approximation polynomials of different order. Then, the error is estimated by local extrapolation.

With an error estimator based on *Milne*-device or local extrapolation, the communication-step size can easily be controlled by using well-known step-size controllers, for instance, a PI-step size controller.

Appendix: Alternative Co-simulation Approach with Improved Convergence Order

In Sect. 11.5.2, we have seen, that the local error of the position variables converges with order $k + 3$. However, the velocity variables converge only with order $k + 2$, which causes that the global error converges only with order $k + 1$. Combining the two methods described in Sect. 11.4 yields an alternative method (method 3), which may increase the convergence order of the local error of the velocity variables. Thus, the local error may converge with order $k + 3$ and consequently the global error may converge with order $k + 2$. We define the new polynomial

$$\mathbf{p}_{N+1}^\alpha(t) = \alpha \mathbf{p}_{N+1}^{\text{int}}(t) + (1 - \alpha) \mathbf{p}_{N+1}^{\text{ext}}(t), \quad (11.55)$$

where α is an arbitrary real number. For analyzing the consistency, we define

$$\mathbf{y}_{N+1}^\alpha := \mathbf{y}(T_N) + \int_{T_N}^{T_{N+1}} \mathbf{F}^{\text{co}}(t, \mathbf{y}(t), \mathbf{p}_{N+1}^\alpha(t)) dt.$$

We have

$$\begin{aligned} \mathbf{y}_{N+1}^\alpha - \mathbf{y}(T_{N+1}) &= \int_{T_N}^{T_{N+1}} \mathbf{F}^{\text{co}}(t, \mathbf{y}(t), \mathbf{p}_{N+1}^\alpha(t)) - \mathbf{F}^{\text{co}}(t, \mathbf{y}(t), \mathbf{p}(t)) dt \\ &= \int_{T_N}^{T_{N+1}} \mathbf{F}_p^{\text{co}}(t, \mathbf{y}(t), \mathbf{p}(t)) (\mathbf{p}_{N+1}^\alpha(t) - \mathbf{p}(t)) dt + \mathcal{O}(H^{k+3}) \\ &= \mathbf{F}_p^{\text{co}}(T_N, \mathbf{y}_N, \mathbf{u}_N) \int_{T_N}^{T_{N+1}} \mathbf{p}_{N+1}^\alpha(t) - \mathbf{p}(t) dt + \mathcal{O}(H^{k+3}) \\ &= \mathbf{F}_p^{\text{co}}(T_N, \mathbf{y}_N, \mathbf{u}_N) \int_{T_N}^{T_{N+1}} \mathbf{p}_{N+1}^\alpha(t) - \hat{\mathbf{p}}(t) dt + \mathcal{O}(H^{k+3}). \end{aligned} \quad (11.56)$$

With the constants

$$D_{N+1}^i := \int_{T_N}^{T_{N+1}} L_{N+1}^i(t) dt \quad (i \in \{k, k + 1\}) \quad (11.57)$$

and with Eqs. (11.41) and (11.42), we obtain

$$\int_{T_N}^{T_{N+1}} \mathbf{p}_{N+1}^\alpha(t) - \hat{\mathbf{p}}(t) dt = D_{N+1}^{k+1} - \alpha D_{N+1}^k, \quad (11.58)$$

which vanishes for $\alpha = \frac{D_{N+1}^{k+1}}{D_{N+1}^k}$. Together with Eq. (11.56), we get

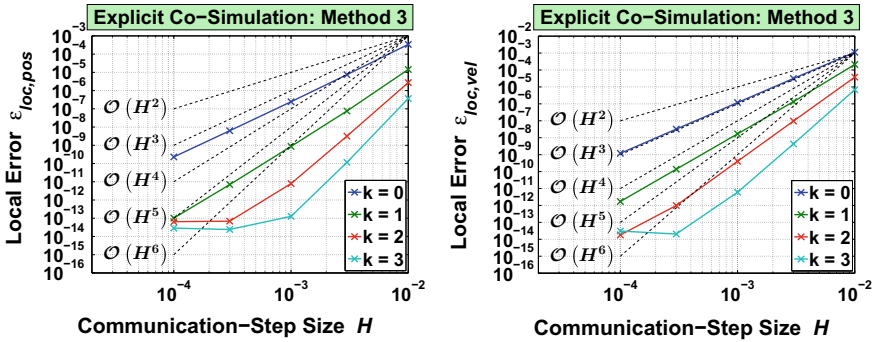


Fig. 11.15 Convergence of the local error of method 3 on position and velocity level

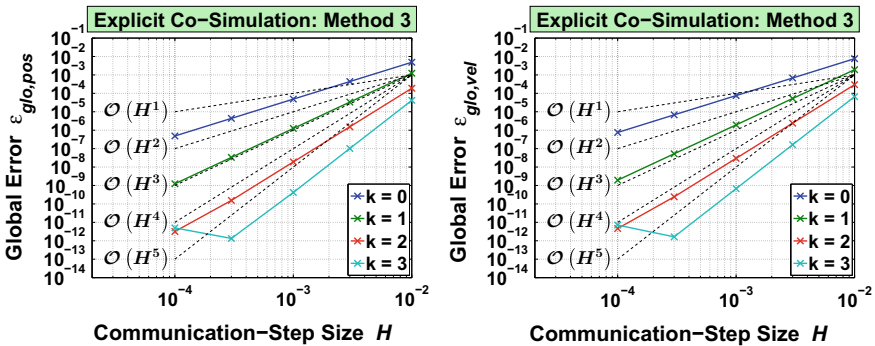


Fig. 11.16 Convergence of the global error of method 3 on position and velocity level

$$\frac{1}{H} \|\mathbf{y}_{N+1}^\alpha - \mathbf{y}(T_{N+1})\| = \mathcal{O}(H^{k+2}). \tag{11.59}$$

Figure 11.15 shows the convergence of the local error on position and velocity level for method 3. The convergence plots confirm that the local error converges with order $k + 3$ on position and on velocity level. As expected, the global error converges with order $k + 2$, which can be seen in Fig. 11.16. To derive an error estimator, it seems straightforward to use method 1 as reference. Then, the error is estimated by

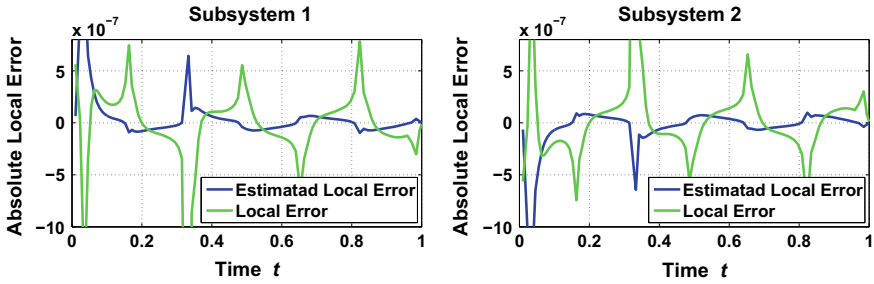


Fig. 11.17 Comparison of the estimated local error to the real local error of method 3 carried out with quadratic approximation polynomials ($k = 2$) and predicted values $\mathbf{u}_{N+1}^{\text{prc}}$ calculated with 4 sampling points; reference solution is carried out with method 1

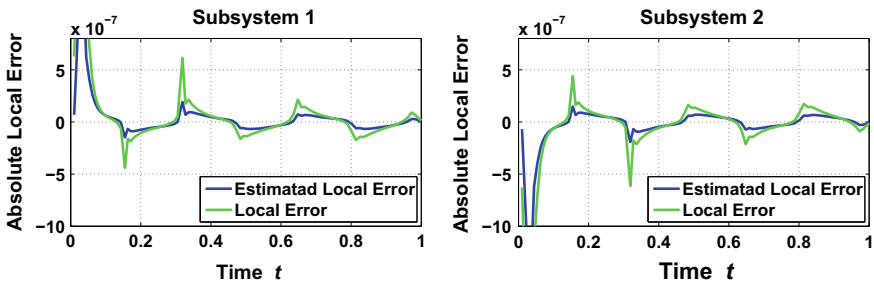


Fig. 11.18 Comparison of the estimated local error to the real local error of method 3 carried out with quadratic approximation polynomials ($k = 2$) and predicted values $\mathbf{u}_{N+1}^{\text{prc}}$ calculated with 5 sampling points; reference solution is carried out with method 1

$$\hat{\varepsilon}_{N+1}^\alpha := \frac{1}{\alpha} \left(1 - \frac{C_{N+1}^{k+1}}{C_{N+1}^k} \right) \|\mathbf{q}_{N+1}^\alpha - \mathbf{q}_{N+1}^{\text{ext}}\|, \tag{11.60}$$

where \mathbf{q}_{N+1}^α is the numerical solution of the position variables of the simulations carried out with the polynomials $\mathbf{p}_{N+1}^\alpha(t)$. Unfortunately, the local error is too badly estimated, if the predicted values $\mathbf{u}_{N+1}^{\text{prc}}$ (see Sect. 11.4.2) are computed with only $k + 2$ sampling points. A similar problem was already mentioned in Remark 11.2 (Sect. 11.7.2). Figure 11.17 shows the local error and the estimated local error. The simulations have been carried out with quadratic approximation polynomials ($k = 2$). The predicted coupling values are extrapolated with four supporting points.

To improve the error estimator, method 3 is modified by increasing the extrapolation order for calculating the predicted coupling values. As can be seen in Fig. 11.18, the error estimator is improved, if the predicted values are extrapolated by five sampling point.

References

1. Eich-Soellner, E., Führer, C.: Numerical Methods in Multibody Dynamics, vol. 45. Teubner, Stuttgart (1998)
2. Gomes, C., et al.: Co-simulation: state of the art. [arXiv:1702.00686](https://arxiv.org/abs/1702.00686) (2017)
3. Gustafsson, K., Lundh, M., Söderlind, G.: API stepsize control for the numerical solution of ordinary differential equations. BIT Numer. Math. **28**(2), 270–287 (1988)
4. Kübler, R., Schiehlen, W.: Two Methods of Simulator Coupling. Math. Comput. Model. Dyn. Syst. **6**(2), 93–113 (2000)
5. Meyer, T., Li, P., Lu, D., Schweizer, B.: Implicit co-simulation method for constraint coupling with improved stability behavior. Multibody Syst. Dyn. 1–27 (2018)
6. Milne, W. E.: Numerical integration of ordinary differential equations. Am. Math. Mon. **33**(9), 455–460 (1926)
7. Schweizer, B., Li, P., Lu, D.: Explicit and implicit cosimulation methods: stability and convergence analysis for different solver coupling approaches. J. Comput. Nonlinear Dyn. **10**(5), 051007 (2015)

Chapter 12

Stability and Error Analysis of Applied-Force Co-simulation Methods Using Mixed One-Step Integration Schemes



Bryan Olivier, Olivier Verlinden and Georges Kouroussis

Abstract Co-simulation schemes are designed to couple subsystems during the integration process. Therefore, any complex or multi-physics system can be split into subsystems in its mathematical representation, and re-coupled using a co-simulation scheme. Dealing separately with each subsystem, its own characteristics and specifically its own solver is the purpose of this decoupling/re-coupling mechanism. Before making a choice between all the existing solver-coupling schemes for a complex mechanical system, it is interesting to know which one is the most efficient. Therefore, this paper studies the performance of the Jacobi and Gauß–Seidel methods using one-step integration schemes applied on a double harmonic oscillator. However, since most of the mechanical joints generate elastic forces, the study concerns applied-force schemes only.

12.1 Introduction

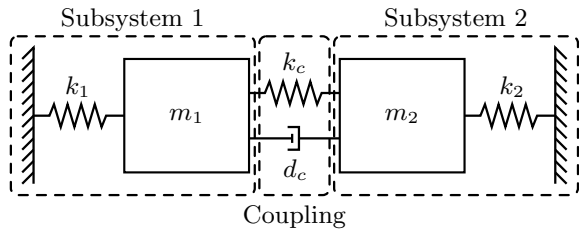
To couple two (or more) dynamic models, it is often better to consider each model in the appropriate approach, to avoid a single and rigorous analysis and to prefer a decoupled approach. For example, in order to predict ground vibrations induced by railway vehicles, it is common to develop decoupled vehicle/track/soil models [1–3]. The model proposed by Kouroussis et al. [4] is a sequential two-step model that uses a multibody modeling approach for the vehicle/track subsystem (including

B. Olivier (✉) · O. Verlinden · G. Kouroussis
Faculty of Engineering, Department of Theoretical Mechanics,
Dynamics and Vibrations, University of Mons, Place du Parc 20, 7000 Mons, Belgium
e-mail: bryan.olivier@umons.ac.be

O. Verlinden
e-mail: olivier.verlinden@umons.ac.be

G. Kouroussis
e-mail: georges.kouroussis@umons.ac.be

Fig. 12.1 Double Harmonic Oscillator with masses m_i , stiffnesses k_i and damping coefficient d_c



also a reduced model of the soil) for the first step and, for the second step, a finite element model for the soil to simulate ground vibration propagation. This two-step simulation process using different solvers could be transformed into a co-simulation process. Therefore, it is important to determine where the global model (vehicle-rails-sleepers-ballast-soil) must be split and which co-simulation scheme provides the best results. Since each section where the system could be split generates elastic constraints, only applied-forces co-simulation schemes will be considered.

This study presents an error analysis and also investigates the numerical stability of two co-simulation schemes: the Jacobi and Gauß–Seidel schemes [5] applied to a double harmonic oscillator with different stiffness ratios (Fig. 12.1). Research in this field was already performed by Busch [5], however, this paper proposes an alternative way to build the amplification matrices [6] of co-simulated mechanical subsystems with their respective integration scheme. Furthermore, the impact of the solver used for each subsystem and the coupling effect of two different solvers are also investigated.

Figure 12.1 represents the system studied in this paper. Separated, the subsystems i (composed of a mass m_i and connected to the reference body with a spring of stiffness k_i where $i = 1, 2$) are supposed to be undamped in order to distinguish the numerical damping caused by both numerical integration and coupling schemes. The coupling joint (represented by k_c and d_c), however, presents damping to limit instabilities. Since the link between subsystem 1 and subsystem 2 is flexible, both systems are re-coupled by forces. The choice of applied-force coupling schemes has two main interests:

- Any native elastic joint (even with damping) can be used to split an entire system.
- Rigid joints can be approximated by increasing the stiffness of the link (k_c) in the model (when $k_c \rightarrow \infty$, a situation close to the gluing is created but since the value of the stiffness has to be defined, the method does not include algebraic constraints).

Furthermore, different stiffness ratios are studied to illustrate the performance of Jacobi and Gauß–Seidel co-simulation schemes in different situations representing different physical phenomena.

12.2 Amplification Matrix Construction

The amplification matrix that defines the modification of the state variables over a time step has interesting properties that qualify the numerical schemes used to integrate a given system. Before analyzing its properties, the following lines describe how to construct that matrix. The method described here is comparable with the method proposed by Busch [5].

Considered separately, each subsystem i consists of a harmonic oscillator with its own eigenfrequency $\omega_{0i} = \sqrt{\frac{k_i}{m_i}}$. Therefore, the equation of motion of this system is

$$\ddot{y}_i + \omega_{0i}^2 y_i = 0 \quad (12.1)$$

which is equivalent, for first integration schemes, to

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \dot{\mathbf{x}}_i + \begin{bmatrix} 0 & -1 \\ \omega_{0i}^2 & 0 \end{bmatrix} \mathbf{x}_i = 0 \quad \text{if} \quad \mathbf{x}_i = \begin{Bmatrix} y_i \\ \dot{y}_i \end{Bmatrix}. \quad (12.2)$$

Considering a one-step and first-order numerical integration formula \mathbf{A} , the discretized system becomes

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \dot{\mathbf{x}}_i^{t+h} + \begin{bmatrix} 0 & -1 \\ \omega_{0i}^2 & 0 \end{bmatrix} \mathbf{x}_i^{t+h} = 0 \quad (12.3a)$$

$$\mathbf{A}(\mathbf{x}_i^t, \dot{\mathbf{x}}_i^t, \mathbf{x}_i^{t+h}, \dot{\mathbf{x}}_i^{t+h}) = 0 \quad (12.3b)$$

and can be re-written in the form

$$\mathbf{P}_i \mathbf{z}_i^{t+h} + \mathbf{Q}_i \mathbf{z}_i^t = 0 \quad (12.4)$$

where $\mathbf{z}_i^{t+h} = \{\mathbf{x}_i^{t+h} \dot{\mathbf{x}}_i^{t+h}\}^T$ and $\mathbf{z}_i^t = \{\mathbf{x}_i^t \dot{\mathbf{x}}_i^t\}^T$.

It can be noticed that Eq. 12.4 can be written for any mechanical system with n_{cp} configuration parameters and leads to $4n_{cp}$ equations (for mechanical systems described by a first order formulation). Furthermore, it yields the discretized state space representation

$$\mathbf{z}_i^{t+h} = -\mathbf{P}_i^{-1} \mathbf{Q}_i \mathbf{z}_i^t = \mathbf{A}_i^0 \mathbf{z}_i^t \quad (12.5)$$

where \mathbf{A}_i^0 is the amplification matrix of the uncoupled subsystem i .

12.2.1 Amplification Matrix of Co-simulation Schemes

As for classical integration schemes, the evolution of the state variables (configuration parameters and their first time derivative for mechanical systems) over a time step can be expressed, for a given co-simulation scheme, as:

$$\mathbf{z}^{t+h} = \mathbf{A}^{CS} \mathbf{z}^t \quad (12.6)$$

representing the relationship between the state variables at time t and $t + h$ due to the integration scheme. The coupling amplification matrix \mathbf{A}^{CS} is a function of the time step, both subsystems, the integration schemes used and, furthermore, the coupling scheme used. For both subsystems, the discretized state space representation becomes, to take the coupling into account,

$$\mathbf{P}_i \mathbf{z}_i^{t+h} + \mathbf{Q}_i \mathbf{z}_i^t + \mathbf{R}_i \mathbf{u}_i = 0 \quad (12.7)$$

in which:

- \mathbf{R}_i matrix has a size of $4n_{cp} \times n_{cv}$ and defines how the subsystems are interacting (with n_{cv} , the number of coupling variables). This matrix could also be used in the monolithic system (in opposition to the co-simulated system) in order to define the interaction between both equations of the entire system.
- \mathbf{u}_i vector has a length of $n_{cv} \times 1$ and defines the coupling variables chosen, for each subsystem, with respect to the state variables of the others. This vector defines the coupling scheme used to perform the integration.

In general, for both subsystems, the inputs are developed in,

$$\mathbf{u}_i = \mathbf{U}_i^t \mathbf{z}_{j \neq i}^t + \mathbf{U}_i^{t+h} \mathbf{z}_{j \neq i}^{t+h} \quad (12.8)$$

where \mathbf{U}_i^t and \mathbf{U}_i^{t+h} defines the connexion scheme between both subsystems and, among other things, the order in which both subsystems are integrated and how the coupling variables are predicted for the second integration when the scheme is sequential. In particular, in our system without any link damping ($d_c = 0 \rightarrow n_{cv} = 1$), taking into account that $\mathbf{z}_1^t = \{y_1^t \dot{y}_1^t \ddot{y}_1^t\}^T$ and $\mathbf{z}_2^t = \{y_2^t \dot{y}_2^t \ddot{y}_2^t\}^T$ (\dot{y}_1^t and \dot{y}_2^t are taken twice into account due to the first order transformation of the second order equations of motion), the following cases can happen:

- $\mathbf{U}_1^t = [1 \ 0 \ 0 \ 0]$, $\mathbf{U}_1^{t+h} = [0 \ 0 \ 0 \ 0]$, $\mathbf{U}_2^t = [1 \ 0 \ 0 \ 0]$ and $\mathbf{U}_2^{t+h} = [0 \ 0 \ 0 \ 0]$ correspond to the Jacobi scheme (illustrated in Fig. 12.2b) in which each integration does not influence each other over the same macro-time step.
- $\mathbf{U}_1^t = [1 \ 0 \ 0 \ 0]$, $\mathbf{U}_1^{t+h} = [0 \ 0 \ 0 \ 0]$, $\mathbf{U}_2^t = [0 \ 0 \ 0 \ 0]$ and $\mathbf{U}_2^{t+h} = [1 \ 0 \ 0 \ 0]$ correspond to the Gauß–Seidel scheme (illustrated in Fig. 12.2a) where the first subsystem is integrated without any interaction with subsystem 2 (such as in Jacobi scheme) but the second subsystem is integrated using the output of the first integration. This scheme is completely sequential.
- $\mathbf{U}_1^t = [0 \ 0 \ 0 \ 0]$, $\mathbf{U}_1^{t+h} = [1 \ 0 \ 0 \ 0]$, $\mathbf{U}_2^t = [1 \ 0 \ 0 \ 0]$ and $\mathbf{U}_2^{t+h} = [0 \ 0 \ 0 \ 0]$ lead to the same as the previous scheme but in which subsystem 2 is integrated before subsystem 1.
- $\mathbf{U}_1^t = [0 \ 0 \ 0 \ 0]$, $\mathbf{U}_1^{t+h} = [1 \ 0 \ 0 \ 0]$, $\mathbf{U}_2^t = [0 \ 0 \ 0 \ 0]$ and $\mathbf{U}_2^{t+h} = [1 \ 0 \ 0 \ 0]$ correspond to a scheme where each subsystem is integrated before the other one which is practically impossible. Using Eqs. 12.7 and 12.8 with that schemes yields the amplification matrix of the monolithic system. However, even if it is possible with

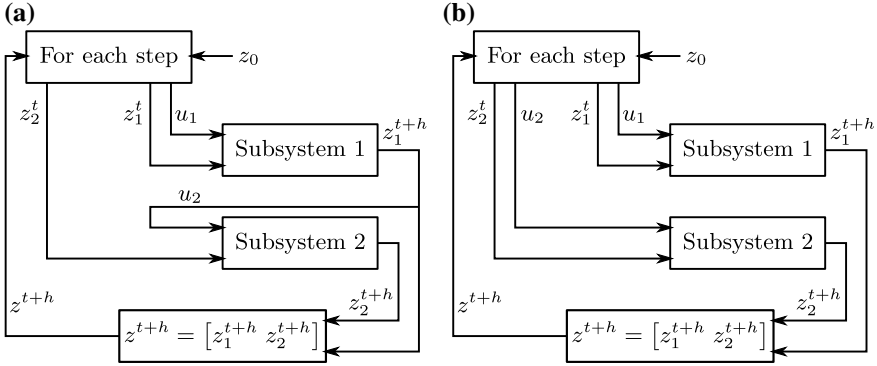


Fig. 12.2 Gauß–Seidel (2a) and Jacobi (2b) co-simulation schemes: step procedure with initial conditions z_0 , subsystem i state variables z_i and subsystem i inputs u_i

that simple example, iteration becomes necessary when working with dedicated software packages for each subsystem.

To obtain those matrices for the system with a link damping ($d_c \neq 0 \rightarrow n_{cv} = 2$), a second line must be added with a 1 in the third column (or second due to the transformation of second order mechanical equations in first order equations) to select the speed as a second coupling variable.

For two subsystems, Eqs. 12.7 and 12.8 become

$$z_1^{t+h} + P_1^{-1} Q_1 z_1^t + P_1^{-1} R_1 (U_1^t z_2^t + U_1^{t+h} z_2^{t+h}) = 0 \quad (12.9a)$$

$$z_2^{t+h} + P_2^{-1} Q_2 z_2^t + P_2^{-1} R_2 (U_2^t z_1^t + U_2^{t+h} z_1^{t+h}) = 0 \quad (12.9b)$$

in which $P_2^{-1} Q_2$ and $P_1^{-1} Q_1$ are, using Eq. 12.5, the opposite of the amplification matrices $-A_1^0$ and $-A_2^0$ of each uncoupled subsystem. This system yields the matrix form

$$\begin{bmatrix} \mathbf{I} & P_1^{-1} R_1 U_1^{t+h} \\ P_2^{-1} R_2 U_2^{t+h} & \mathbf{I} \end{bmatrix} \begin{Bmatrix} z_1^{t+h} \\ z_2^{t+h} \end{Bmatrix} + \begin{bmatrix} -A_1^0 & P_1^{-1} R_1 U_1^t \\ P_2^{-1} R_2 U_2^t & -A_2^0 \end{bmatrix} \begin{Bmatrix} z_1^t \\ z_2^t \end{Bmatrix} = 0 \quad (12.10)$$

that could be shortened in the generic form

$$P^{CS} z^{t+h} + Q^{CS} z^t = 0 \quad (12.11)$$

if $z^{t,t+h} = \{z_1^{t,t+h} z_2^{t,t+h}\}^T$. Finally, such as for classical monolithic systems, the amplification matrix of co-simulation methods is expressed by

$$A^{CS} = - (P^{CS})^{-1} Q^{CS} \quad (12.12)$$

where \mathbf{P}^{CS} and \mathbf{Q}^{CS} are defined by Eqs. 12.10 and 12.11 where \mathbf{I} is the identity matrix with a size tuned to the number of variables \mathbf{z}_i of subsystem i scheme.

A few remarks can be made on Eq. 12.10:

- The coupling between both subsystems clearly appears through the non-diagonal terms. Indeed, if both \mathbf{R}_i are matrices filled with zeros, both systems appear to be fully decoupled and the amplification matrix contains only both uncoupled amplification matrices \mathbf{A}_1^0 and \mathbf{A}_2^0 .
- When \mathbf{U}_i^{t+h} is filled with zeros, the corresponding subsystem is not explicitly dependent upon the second subsystem integrated states variables.
- When both \mathbf{U}_i^t are filled with zeros, the amplification matrix obtained corresponds to the monolithic system matrix. As specified earlier, this scheme is practically unreachable with a co-simulation scheme without infinite iterations.

12.2.2 Amplification Matrix of Iterated Schemes

When the accuracy of the solution is more important than the computation time, the co-simulation scheme can be iterated in order to improve accuracy and numerical stability. Indeed, in Eq. 12.7, the closer \mathbf{u}_i and $\mathbf{z}_{j \neq i}^{t+h}$, the closer the co-simulation and the monolithic schemes should be. Hence, for a given coupling scheme, a first integration of both subsystems could be performed to have a better estimation of \mathbf{u}_i values in a second full integration.

Taking into account iterations in the co-simulation scheme, the state space representation of a subsystem scheme, Eq. 12.7 becomes, for the k th iteration of a given scheme:

$$\mathbf{P}_i \mathbf{z}_i^{t+h,k} + \mathbf{Q}_i \mathbf{z}_i^t + \mathbf{R}_i \mathbf{u}_i^k = 0 \quad (12.13)$$

where $\mathbf{z}_i^{t+h,k}$ is the k th iteration of the state variables and \mathbf{u}_i^k is the k th iteration of the inputs of subsystem i . These inputs are obtained by modifying adequately Eq. 12.8:

$$\mathbf{u}_i^k = \mathbf{U}_i^{s,k} \mathbf{z}_{j \neq i}^{t+h,k} + \mathbf{U}_i^{s-1,k} \mathbf{z}_{j \neq i}^{t+h,k-1} \quad (12.14)$$

where, such as for non-iterated systems, the matrices $\mathbf{U}_i^{s,k}$ and $\mathbf{U}_i^{s-1,k}$ define the procedure followed to integrate both subsystems during the iteration process. Indeed, if k represents the number of iterations, $\mathbf{U}_i^{s,k}$ defines whether the integration of subsystem i takes into account an estimation of the state variables $\mathbf{z}_{j \neq i}^{t+h,k}$ resulting from an integration of the other subsystem during the same k th iteration. However, $\mathbf{U}_i^{s,k}$ defines whether the integration of subsystem i takes into account the estimation of the state variables $\mathbf{z}_{j \neq i}^{t+h,k-1}$ resulting from the previous iteration $k-1$ th. It must be remarked that an extrapolation of the state variables obtained in N previous iterations could lead to a faster convergence to the monolithic scheme. In this study, the iteration matrices are defined as follows (for the undamped coupling scheme):

- $\mathbf{U}_1^{s,k} = [0\ 0\ 0\ 0]$, $\mathbf{U}_1^{s,k-1} = [1\ 0\ 0\ 0]$, $\mathbf{U}_2^{s,k} = [0\ 0\ 0\ 0]$ and $\mathbf{U}_2^{s,k-1} = [1\ 0\ 0\ 0]$ which leads to a iterated-Jacobi scheme or

- $\mathbf{U}_1^{s,k} = [0\ 0\ 0\ 0]$, $\mathbf{U}_1^{s,k-1} = [1\ 0\ 0\ 0]$, $\mathbf{U}_2^{s,k} = [1\ 0\ 0\ 0]$ and $\mathbf{U}_2^{s,k-1} = [0\ 0\ 0\ 0]$ which leads to a iterated-Gauß-Seidel scheme;
- these matrices could vary between two successive iterations.

Considering Eqs. 12.13 and 12.14, for two subsystems solved by an iterated co-simulation scheme, it can be written

$$\mathbf{P}_1 \mathbf{z}_1^{t+h,k} + \mathbf{Q}_1 \mathbf{z}_1^t + \mathbf{R}_1 \left(\mathbf{U}_1^{s,k} \mathbf{z}_2^{t+h,k} + \mathbf{U}_1^{s-1,k} \mathbf{z}_2^{t+h,k-1} \right) = 0 \quad (12.15a)$$

$$\mathbf{P}_2 \mathbf{z}_2^{t+h,k} + \mathbf{Q}_2 \mathbf{z}_2^t + \mathbf{R}_2 \left(\mathbf{U}_2^{s,k} \mathbf{z}_1^{t+h,k} + \mathbf{U}_2^{s-1,k} \mathbf{z}_1^{t+h,k-1} \right) = 0 \quad (12.15b)$$

which can be expressed in the following matrix form

$$\begin{bmatrix} \mathbf{P}_1 & \mathbf{R}_1 \mathbf{U}_1^{s,k} \\ \mathbf{R}_2 \mathbf{U}_2^{s,k} & \mathbf{P}_2 \end{bmatrix} \begin{Bmatrix} \mathbf{z}_1^{t+h,k} \\ \mathbf{z}_2^{t+h,k} \end{Bmatrix} + \begin{bmatrix} \mathbf{Q}_1 & 0 \\ 0 & \mathbf{Q}_2 \end{bmatrix} \begin{Bmatrix} \mathbf{z}_1^t \\ \mathbf{z}_2^t \end{Bmatrix} + \begin{bmatrix} 0 & \mathbf{R}_1 \mathbf{U}_1^{s,k-1} \\ \mathbf{R}_2 \mathbf{U}_2^{s,k-1} & 0 \end{bmatrix} \begin{Bmatrix} \mathbf{z}_1^{t+h,k-1} \\ \mathbf{z}_2^{t+h,k-1} \end{Bmatrix} = 0 \quad (12.16)$$

or in a shorter form

$$\mathbf{P}^{it} \mathbf{z}^{t+h,k} + \mathbf{Q}^{it} \mathbf{z}^t + \mathbf{R} \mathbf{U}^{it} \mathbf{z}^{t+h,k-1} = 0 \quad (12.17)$$

Since every estimation is based only on the initial conditions of the step \mathbf{z}^t , the state variables estimation for the k th iteration are expressed as:

$$\mathbf{z}^{t+h,k} = \mathbf{A}^{CS,k} \mathbf{z}^t \quad (12.18)$$

which yields, with Eq. 12.17,

$$\mathbf{A}^{CS,k} = -(\mathbf{P}^{it})^{-1} (\mathbf{Q}^{it} + \mathbf{R} \mathbf{U}^{it} \mathbf{A}^{CS,k-1}) \quad (12.19)$$

with $\mathbf{A}^{CS,0}$ computed using Eq. 12.12.

12.3 Results

The definition of the amplification matrix of co-simulation schemes provides crucial information about the efficiency of the scheme. Indeed, the spectral radius $\rho(\mathbf{A})$ [7], an indicator of the stability of the scheme, can be deduced from the amplification matrix \mathbf{A} . The stability criterion of a numerical scheme is

$$\rho(\mathbf{A}) = \max (|\lambda(\mathbf{A})|) < 1 \quad (12.20)$$

with λ the function computing the eigenvalues of a matrix. However, a stable scheme does not mean that the scheme is accurate. Indeed, more than the stability, the spectral radius ρ provides a global idea on the transformation that the scheme applied on the real results:

- if $\rho = 1$, the numerical scheme is strictly stable which means that there is at least one mode that conserves its energy;
- if $\rho < 1$, the numerical scheme dissipates energy. That characteristic is a synonym of stability;
- if $\rho > 1$ the numerical scheme introduces energy into the integrated system. This causes instability and the higher the spectral radius, the faster the integrated system will diverge from the analytical solution.

In Fig. 12.3, the spectral radii of 5 different schemes are studied, for Gauß-Seidel and Jacobi coupling schemes applied with forward and backward Euler methods.

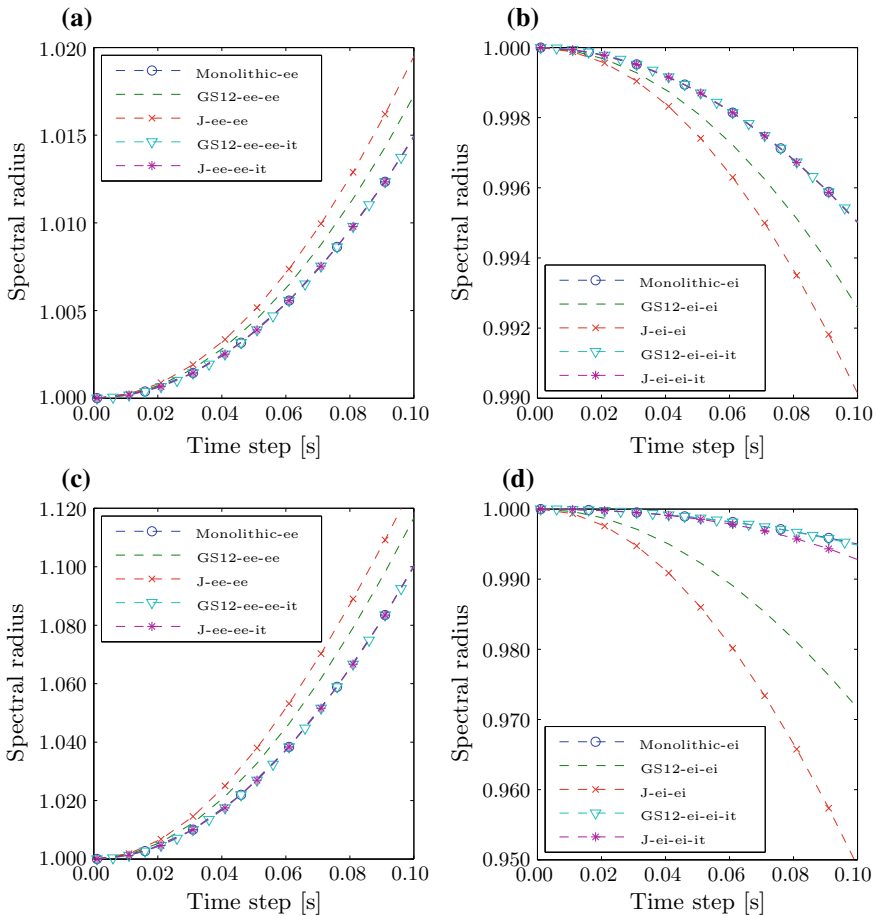


Fig. 12.3 Spectral radii of Gauß-Seidel with the order subsystem 1 before subsystem 2 (GS12) and Jacobi (J) schemes with forward Euler (ee) and backward Euler (ei) integration schemes. (3a) and (3b) are computed for $\frac{k_c}{k_1} = 1$, (3c) and (3d) are computed for $\frac{k_c}{k_1} = 10$ while $\frac{k_2}{k_1} = 1$, $\frac{m_2}{m_1} = 1$ and $d_c = 0$

The latter were chosen for their lack of efficiency in order to illustrate clearly the characteristics of the schemes. In each graph, the reference is assumed to be the monolithic corresponding scheme. A few remarks can be deduced:

- For forward Euler, Figs. 12.3a, c show that the Jacobi scheme is slightly more unstable than the Gauß-Seidel one. Indeed, since Gauß-Seidel is sequential, the second integration input parameters are already a better estimation than in the Jacobi scheme. However, the monolithic scheme provides better results than both coupling schemes.
- For backward Euler, Figs. 12.3b, d show that the Jacobi scheme is slightly more damped than the Gauß-Seidel one which is also more damped than the monolithic scheme. The reason is identical as above.
- Between Figs 12.3a–d, the coupling stiffness ratio $\frac{k_c}{k_1}$ was multiplied by 10. It can be noticed that the stiffer the coupling, the less accurate the results are.
- In each graph, a 1-iteration version of both schemes is studied using Eq. 12.19. Generally, the results are better than each non-iterated corresponding scheme. It can also be proven that, for explicit Euler schemes, it converges to the corresponding monolithic scheme in a single iteration for both Jacobi and Gauß-Seidel coupling methods.

More than the spectral radius, the frequency error and the damping ratios can be computed using the amplification matrix of a scheme. Those parameters are defined using the continuous equivalent λ_i^c of the discretized form of the eigenvalues λ_i of the amplification matrix \mathbf{A} given in Eq. 12.21:

$$\lambda_i^c = \frac{\ln \lambda_i}{h} = \sigma_i \pm j\omega_i \quad (12.21)$$

with h the timestep. From this definition, the damping ratio ξ_{ω_i} and the frequency error ϵ_{ω_i} of mode i can be computed using Eqs. 12.22a and 12.22b:

$$\xi_{\omega_i} = \frac{-\sigma_i}{\sqrt{\sigma_i^2 + \omega_i^2}} \quad (12.22a)$$

$$\epsilon_{\omega_i} = \frac{\sqrt{\sigma_i^2 + \omega_i^2} - \omega_{0i}}{\omega_{0i}} \quad (12.22b)$$

with ω_{0i} the analytical eigenfrequency of mode i . Figure 12.4 illustrates these parameters for mixed forward/backward Euler schemes taking into account that backward Euler is always applied on the second subsystem. Since Gauß-Seidel seemed to provide the best performance, the situations in which both subsystems are integrated using forward and backward Euler methods, with Gauß-Seidel coupling, are taken as references to compare the mixed schemes with. It can be observed that:

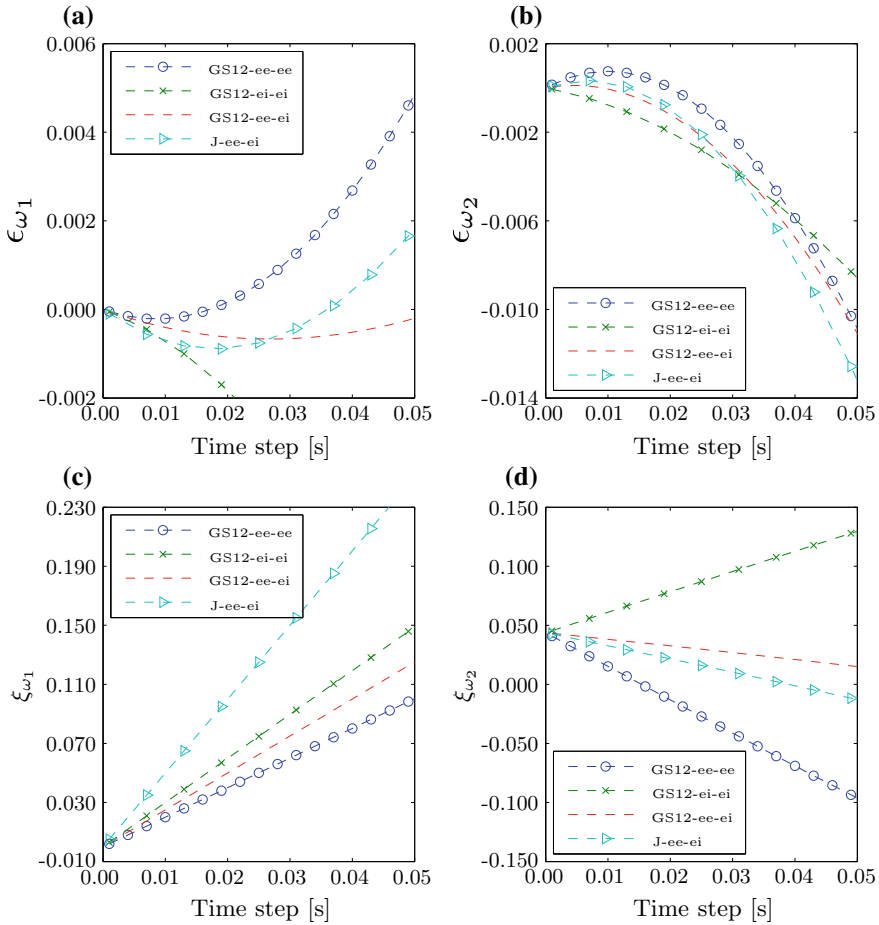


Fig. 12.4 Relative frequency error (ϵ_{ω_j}) and damping ratio (ξ_{ω_j}) of Gauss-Seidel with the order subsystem 1 before subsystem 2 (GS12) and Jacobi (J) schemes with mixed forward Euler (ee) and backward Euler (ei) integration schemes. (4a) and (4c) concern the first eigenfrequency ω_1 and (4b) and (4d) concern the second eigenfrequency ω_2 with $\omega_1 < \omega_2$ while $\frac{k_c}{k_1} = 10$, $\frac{k_2}{k_1} = 1$, $\frac{m_2}{m_1} = 1$ and $d_c = 0.2$

- for small time steps, the frequency error of the second eigenfrequency is smaller in mixed schemes than in fully explicit/implicit schemes. However, the fully explicit scheme seems to provide a smaller frequency error for the first eigenfrequency;
- both mixed schemes provide a damping closer to 0 for the second eigenfrequency. However, for the first eigenfrequency, the fully explicit scheme provides, once again, a smaller damping;
- both frequency errors converge to 0 with the time step;
- both damping coefficients converge to the physical damping coefficient which means that the numerical damping converges to 0.

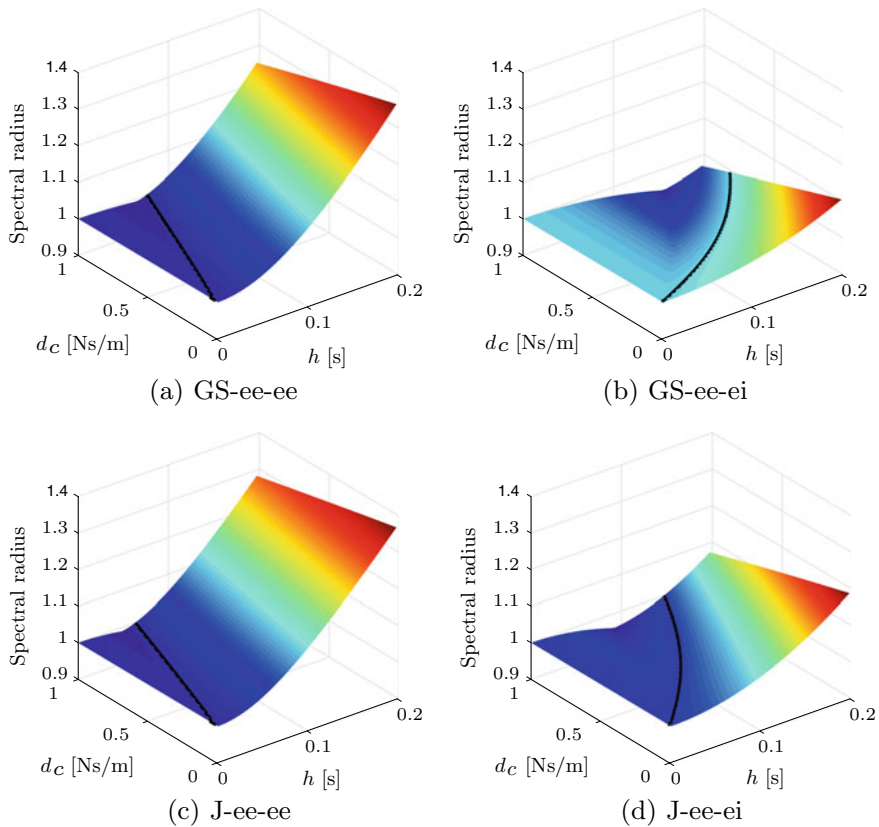


Fig. 12.5 Gauß-Seidel (a, b) and Jacobi (c, d) schemes applied with different integration schemes with $\frac{m_2}{m_1} = 1$, $\frac{k_2}{k_1} = 1$ and $\frac{k_c}{k_1} = 10$. **a** and **c** represents forward Euler (ee) for both subsystems and **b** and **d** represents forward Euler (ee) for subsystem 1 and backward Euler (ei) for subsystem 2. The stability and instability regions are separated by a line at which the spectral radius is 1

Figure 12.5 shows the impact of the link damping d_c on the stability of mixed Jacobi and Gauß-Seidel schemes through their spectral radii. It turns out that:

- both mixed schemes provide spectral radii closer to 1 than the corresponding fully explicit scheme;
- once again, the Gauß-Seidel scheme appears to be better than the Jacobi scheme;
- both mixed scheme offer a larger stability region that grows with the link damping d_c . However, the price for this larger stability region is a larger damping of the solution for a same time step.

12.4 Conclusions

Such as for classic monolithic mathematical representation of mechanical system, the amplification matrix of co-simulated schemes can be written and interesting properties (such as the spectral radius, the frequency error and the damping ratio) can be deduced:

- each coupling scheme used exhibits a zero-stable behavior;
- co-simulation schemes are less accurate than the corresponding monolithic method;
- the Jacobi scheme usually provides a reduced accuracy in comparison with the Gauß-Seidel scheme. This phenomenon is explained by their respective parallel and sequential behavior [5];
- the results provided by iterated schemes are usually better than the corresponding non-iterated schemes. Once again, the sequential Gauß-Seidel scheme appears to give the best results;
- mixed integration schemes produce larger stability regions for the time step choice. However, in return, the damping induced by those methods is larger for a same time step.

References

1. Fernández Ruiz, J., Alves Costa, P., Calada, R., Medina Rodríguez, L.E., Colao, A.: Study of ground vibrations induced by railway traffic in a 3D FEM model formulated in the time domain: experimental validation. *Struct. Infrastruct. Eng.* **13**, 652–664 (2017)
2. Gardien, W., Stuit, H.G.: Modelling of soil vibrations from railway tunnels. *J. Sound Vib.* **267**, 605–619 (2003)
3. Olivier, B., Connolly, D.P., Alves Costa, P., Kouroussis, G.: The effect of embankment on high speed rail ground vibrations. *Int. J. Rail Transp.* **4**, 229–246 (2016)
4. Kouroussis, G., Verlinden, O.: Prediction of railway ground vibrations: accuracy of a coupled lumped mass model for representing the track/soil interaction. *Soil Dyn. Earthq. Eng.* **69**, 220–226 (2015)
5. Busch, M.: Zur effizienten Kopplung von Simulationsprogrammen. Ph.D. Thesis, Kassel University Press (2012)
6. Géradin, M., Rixen, D.: *Mechanical Vibrations: Theory and Application to Structural Dynamics*. Wiley (2015)
7. Hairer, E., Wanner, G.: *Solving Ordinary Differential Equations*. Springer, Berlin (1993)

Chapter 13

A Strategy to Conduct Numerical Simulation of Wind Turbine Considering the Soil-Structure-Interaction by Using a Coupled FEM-SBFEM Approach in Time Domain



Marco Schauer, Francesca Taddei and Sissy Morawietz

Abstract In order to simulate wind turbines under different load scenarios, the computational model should take into account the aerodynamics of the rotor, the flexibility of tower, foundation and soil, transient operational phases and, first and foremost, the interaction of all these aspects. Whenever vibrations are emitted to soil, they induce waves traveling through the ground. Here, the main focus lies on the Soil-Structure-Interaction (SSI) effects on the dynamic behavior of operating wind turbines. The wind turbine with its foundation and the surrounding soil is modeled by a coupled Finite Element Method/Scaled Boundary Finite Element Method approach.

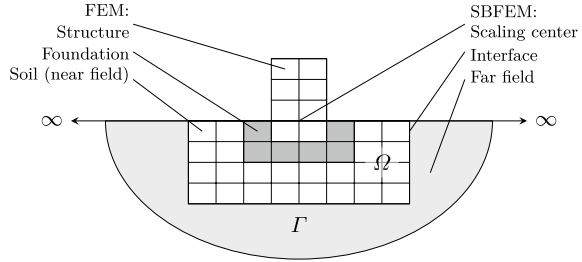
13.1 Introduction

The simulation of the Soil-Structure-Interaction (SSI) for operating wind turbines involves two different mechanical problems: on the one hand, the vibrations due to aerodynamic loads excite the structure of the wind turbine and its foundation and, on the other hand, the vibrations are transferred into the surrounding infinite half-space, which is governed by the Lamé's equations. For such complex problems no analytical or semi-analytical solution is available, so that numerical models are used. The wind turbine can be discretized by using standard methods such as the finite element method. In contrast, the infinite half-space cannot be represented by means

M. Schauer (✉) · S. Morawietz
Institut für Statik, Technische Universität Braunschweig, Beethovenstraße 51,
38106 Braunschweig, Germany
e-mail: m.schauer@tu-braunschweig.de
URL: <https://www.tu-braunschweig.de/statik>

F. Taddei
Chair of Structural Mechanics, Technical University of Munich,
Arcisstrasse 21, 80333 München, Germany
e-mail: francesca.taddei@tum.de
URL: <https://www.bm.bgu.tum.de>

Fig. 13.1 Problem definition [13]



of discrete models, as they are neither able to discretize the infinite domains nor to satisfy the radiation condition to infinity.

To model SSI a sub-structuring method is used and the problem is subdivided into two sub-structures (cf. Fig. 13.1). The structure and its foundation as well as parts of the soil (the near field) are modeled by Finite Element Method (FEM). The infinite half-space (the far field) is discretized by Scaled Boundary Finite Element Method (SBFEM). Both methods are coupled at the common Interface Γ [16].

13.1.1 FEM-SBFEM Coupling

The FEM-SBFEM coupling is described briefly, more information about the theoretical background and its efficient implementation can be found here [12, 14]. The displacement-based FEM in time domain can be derived by the energy theorem

$$-\delta W = - \int_{\Omega} \delta \epsilon^T \sigma d\Omega - \int_{\Omega} \delta \mathbf{u}^T \left(\kappa \frac{d\mathbf{u}}{dt} + \rho \frac{d^2\mathbf{u}}{dt^2} \right) d\Omega \quad (13.1)$$

$$+ \int_{\Omega} \delta \mathbf{u}^T \mathbf{f} d\Omega + \int_{\Gamma} \delta \mathbf{u}^T \mathbf{t} d\Gamma = 0 \quad (13.2)$$

here the vectors \mathbf{u} , $\frac{d\mathbf{u}}{dt} = \dot{\mathbf{u}}$ and $\frac{d^2\mathbf{u}}{dt^2} = \ddot{\mathbf{u}}$ represent displacement, velocity and acceleration, respectively. κ and ρ denote damping ratio and density. Applied tractions and forces are given by \mathbf{t} and \mathbf{f} . After inserting the strain displacement relation $\epsilon = \mathbf{D}\mathbf{u}$ and the stress strain relation $\sigma = \mathbf{E}\epsilon$ in Eq. (13.1) and introducing shape functions the given energy theorem can be rewritten in the matrix-vector form:

$$\begin{bmatrix} \mathbf{M}_{\Omega\Omega} & \mathbf{M}_{\Omega\Gamma} \\ \mathbf{M}_{\Gamma\Omega} & \mathbf{M}_{\Gamma\Gamma} \end{bmatrix} \ddot{\mathbf{u}} + \begin{bmatrix} \mathbf{C}_{\Omega\Omega} & \mathbf{C}_{\Omega\Gamma} \\ \mathbf{C}_{\Gamma\Omega} & \mathbf{C}_{\Gamma\Gamma} \end{bmatrix} \dot{\mathbf{u}} + \begin{bmatrix} \mathbf{K}_{\Omega\Omega} & \mathbf{K}_{\Omega\Gamma} \\ \mathbf{K}_{\Gamma\Omega} & \mathbf{K}_{\Gamma\Gamma} \end{bmatrix} \mathbf{u} = \begin{bmatrix} \mathbf{p}_{\Omega\Omega} \\ \mathbf{p}_{\Gamma\Gamma} \end{bmatrix} \quad (13.3)$$

where \mathbf{M} , \mathbf{C} and \mathbf{K} denotes mass, damping and stiffness matrix, respectively. $\mathbf{p}_{\Omega\Omega}$ and $\mathbf{p}_{\Gamma\Gamma}$ are forces, which are applied either to the structure and its foundation or to the soil, like aerodynamic and earthquake loads. To create the damping matrix, a Rayleigh damping is considered, so that the damping matrix is a combination of

weighted mass matrix and stiffness matrix

$$\mathbf{C} = c_M \mathbf{M} + c_K \mathbf{K}. \quad (13.4)$$

The influence of the infinite half-space is described by SBFEM. The interacting forces at the interface Γ are given by the vector

$$\mathbf{p}_b(t_n) = \gamma \Delta t \mathbf{M}_0^\infty \ddot{\mathbf{u}}_n + \sum_{j=1}^{n-1} \mathbf{M}_{n-j}^\infty (\dot{\mathbf{u}}_j - \dot{\mathbf{u}}_{j-1}), \quad (13.5)$$

here γ is a parameter introduced by the time integration scheme. The unit acceleration impulse matrices \mathbf{M}^∞ are assumed to be constant within one time step Δt they are computed in a pre-process.

The vector of Eq. (13.5) can be added to the right hand side of Eq. (13.3) in order to get a direct and bidirectional coupled FEM-SBFEM formulation

$$\begin{bmatrix} \mathbf{M}_{\Omega\Omega} & \mathbf{M}_{\Omega\Gamma} \\ \mathbf{M}_{\Gamma\Omega} & \mathbf{M}_{\Gamma\Gamma} \end{bmatrix} \ddot{\mathbf{u}} + \begin{bmatrix} \mathbf{C}_{\Omega\Omega} & \mathbf{C}_{\Omega\Gamma} \\ \mathbf{C}_{\Gamma\Omega} & \mathbf{C}_{\Gamma\Gamma} \end{bmatrix} \dot{\mathbf{u}} + \begin{bmatrix} \mathbf{K}_{\Omega\Omega} & \mathbf{K}_{\Omega\Gamma} \\ \mathbf{K}_{\Gamma\Omega} & \mathbf{K}_{\Gamma\Gamma} \end{bmatrix} \mathbf{u} = \begin{bmatrix} \mathbf{p}_{\Omega\Omega} \\ \mathbf{p}_{\Gamma\Gamma} \end{bmatrix} - \begin{bmatrix} \mathbf{0} \\ \mathbf{p}_b \end{bmatrix}. \quad (13.6)$$

The computation in time domain is conducted by executing the generalized- α time stepping algorithm [4].

13.2 Implementation

To analyze the behavior of wind turbines different third-party software packages as well as different in-house computer codes are combined. The general work flow is shown in in Fig. 13.2. Finite element meshes and scaled boundary finite element meshes are generated by using the geometry and mesh generation toolkit CUBIT [11]. Here the geometry is created as well as the final meshes. This information is written into xml-style input files for later usage. To conduct the analysis of the full model of wind turbine its foundation and surrounding soil preliminary steps are required. At first the eigenvalue problem is solved using GNU Octave [6] to compute the eigenvalues and eigenvectors of the wind turbine and its foundation. From the eigenvalues of the system the damping coefficients for the coupled problem are derived. Secondly the response of the infinite half-space is computed, the \mathbf{M}^∞ matrices are stored in compressed binary format. Aerodynamic loads are generated by using FAST [10] from National Renewable Energy Laboratory and written into ASCII files. The process, which solved the coupled FEM-SBFEM computation, has accesses to all previously generated files. The FE-model is build up and the information of the SBFEM-model is used to determine the coupling nodes in order to apply the influence of the infinite half-space to the boundary of the FE-mesh. The aerodynamic loads acting on the rotor and nacelle are applied to the tower top. Additional earthquake loads can be

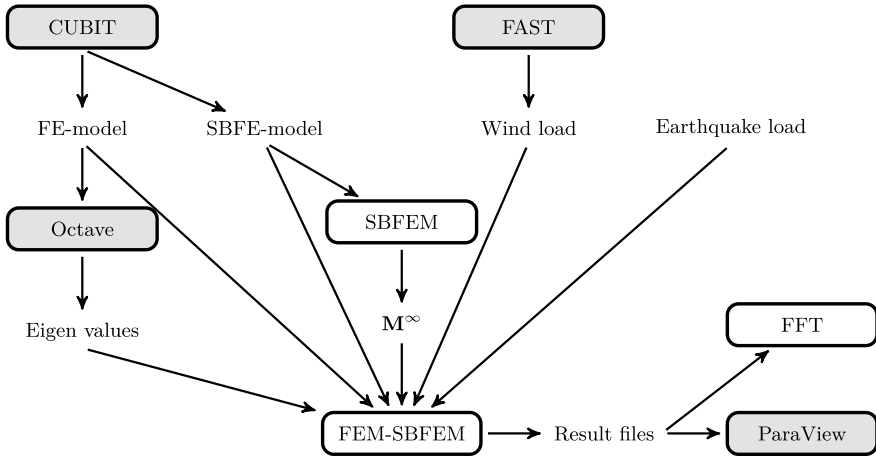


Fig. 13.2 Flow diagram of virtual wind turbine analysis. Boxes filled in gray are commercial or open source software packages

applied if desired. The computation is conducted in time domain, results are written into result files. ParaView [9] is utilized for visualization of the results of the three dimensional model. Time displacement records of chosen nodes can optionally transferred into frequency domain by applying FFT for model assessment and evaluation.

The in-house codes have been implemented carefully to meet the needed requirements. Solving convolution integral is memory and time consuming. In order to handle the high memory consumption and the computational effort to simulate wind turbine and the surrounding soil three dimensionally model reduction techniques and parallel computing techniques are used [12]. Different third party libraries are used to achieve reasonable performance. The program which computes the infinite half-spaces response has to handle dense and sparse matrices as well, that is why ScaLAPACK [3] builds the main core of it. Since the finite element matrices are sparse PETSc [1] builds the main core of the FEM-SBFEM part. The work flow of the FEM-SBFEM coupling looks like

```

parse input file of FEM and SBFEM
build finite element equation system (Eq. (13.3))
initialize interface  $\Gamma$ 
assemble system matrices
while  $t < T$ ,  $t = t + \Delta t$ 
    evaluate far field response (Eq. (13.5))
    solve equation of motion (Eq. (13.6))
    store interface velocity
  
```

in a pseudo program commands. At first all input files are parsed to set up the virtual model. This includes in case of the FEM: nodes, elements, material parameters, damping coefficients, boundary conditions, nodal load and nodal accelerations as well as initial conditions. With this information the description of the FEM part is complete. In case of the SBFEM nodes and elements are processed and M^∞ matrices are read. Afterwards, the interface is initialized and coupling nodes are defined. Then, the system matrices M , K , and D are assembled. These matrices stay constant during the simulation, since linear elastic material behavior and only small displacements of the structure are assumed. The simulation is conducted with a time step length Δt until the total time T is reached. Within this time loop, three major steps are processed. At first the response of the infinite half space is computed. The resulting forces are added to the force vector. After solving the equation of motion, by using a generalized- α time stepping scheme (cf. [4]) the velocities at the interface are stored to use them to evaluate the future time steps.

13.3 Wind Turbine

The design of the numerically investigated wind turbine is given by a reference wind turbine, which has been analyzed by the National Renewable Energy Laboratory (NREL) [8]. This model is chosen for the purpose of validation of the numerical model, since stiffness properties and mass distribution as well as eigen frequencies of the tower, load conditions etc. are known. The wind turbine is a 5-MW plant, with an upwind rotor of 126 m in diameter. The hub height is 90 m. Rotor and Nacelle have a weight of 110 000 kg, 240 000 kg respectively. Both are mounted on a 87.6 m-height steel tower, which has a diameter of 3.87 m on top and 6 m at the bottom. The towers wall thickness changes linearly with increasing height from 0.0351 to 0.0247 m. The tower has a mass of 347 460 kg, so that entire wind turbine has a mass of 697 460 kg, which is founded on a cylindrical foundation. The most important model parameters are summarized in Table 13.1. Rotor and nacelle are considered as a point mass on top of the tower. The cylindrical foundation is made of reinforced concrete with 12% of steel. To model this, it is assumed that the steel is equally distributed within the volume of concrete, so that material properties of reinforcing steel and material properties of the concrete are considered to be smeared. The foundation has a diameter of 16 m, is 2 m thick and it is embedded into the surrounding soil.

13.3.1 Finite Element Model

In order to create a virtual model of the wind tower finite elements are utilized. The numerical model of the tower is developed step by step, to assure correctness of the virtual model. Therefore the finite element model is build up in two steps, starting

Table 13.1 Model parameter of chosen wind tower related to NREL 5-MW [8]

NREL 5-MW	
Rating	5 MW
Rotor orientation	Upwind
Rotor diameter	126 m
Hub diameter	4 m
Hub height	90 m
Tower height	87.6 m
Tower diameter on top/bottom	3.87/6 m
Wall thickness on top/bottom	0.0247/0.0351 m
Rotor mass	110 000 kg
Nacelle mass	240 000 kg
Tower mass	347 460 kg

with the tower. In the second step the foundations and the surrounding soil and the infinite half-space is added. Each model is analyzed by itself to assure accuracy.





13.3.1.1 Tower Modeled as Cantilever

The tower is discretized as a cantilever by using Euler-Bernoulli beam elements, with bending and axial flexibility as well as torsion, so that each nodes provides six degree of freedom (DOF). Masses of rotor and nacelle are introduced by simple lumped-mass elements, which add the additional mass directly to the main diagonal of the mass matrix at corresponding nodes. The tower is discretized by 40 beam elements and therefore 246 DOF. Here the mass is added to the top node of the tower. All six DOF of the bottom node are fixed to satisfy bearing of the cantilever. The side-side and fore-aft direction are analyzed separately without considering coupling effects.

Additionally a more complex model, where coupling effects of side-side and fore-aft direction is considered, is build by using 3312 continuum elements with 6768 nodes, that leads to 20304 DOF. The mass of rotor and nacelle is taken into account by 144 mass elements, which are distributed at the towers very top. In both cases the tower is made of steel using the following material parameters: Young's modulus $E = 2.1 \cdot 10^{11} \text{ N/m}^2$, Poisson's ratio $\nu = 0.3$ and density $\rho = 8500 \text{ kg/m}^3$. The density of steel is increased by 7.65% to take the technical equipment within the tower into account [8].

To assure that the implemented virtual model describes the behavior of the real wind turbine correctly, mass and stiffness properties must be distributed correctly inside the numerical model. Therefore the natural frequencies of the created model are compared with the natural frequencies of a reference model [8]. Table 13.2

Table 13.2 Natural frequencies f [Hz] of the towers model, fixed at its base

Mode					
Euler-bernoulli		1, 2	3, 4	5	6, 7
Angular frequency	ω [rad s ⁻¹]	2.1123	19.3073	49.8095	57.6957
Frequency	f [Hz]	0.3362	3.0729	7.9274	9.1823
Reference solution [8]					
FAST	f [Hz]	0.3240	2.9361		
ADAMS	f [Hz]	0.3164	2.9108		
3-D continuum		1, 2	5, 6	10	13, 14
Angular frequency	ω [rad s ⁻¹]	2.1163	18.8214	49.7607	53.1648
Frequency	f [Hz]	0.3368	2.9955	7.9197	8.4614

summarizes the natural frequencies of the virtual wind turbine discretized by Euler-Bernoulli beam elements and three dimensional continuum elements.

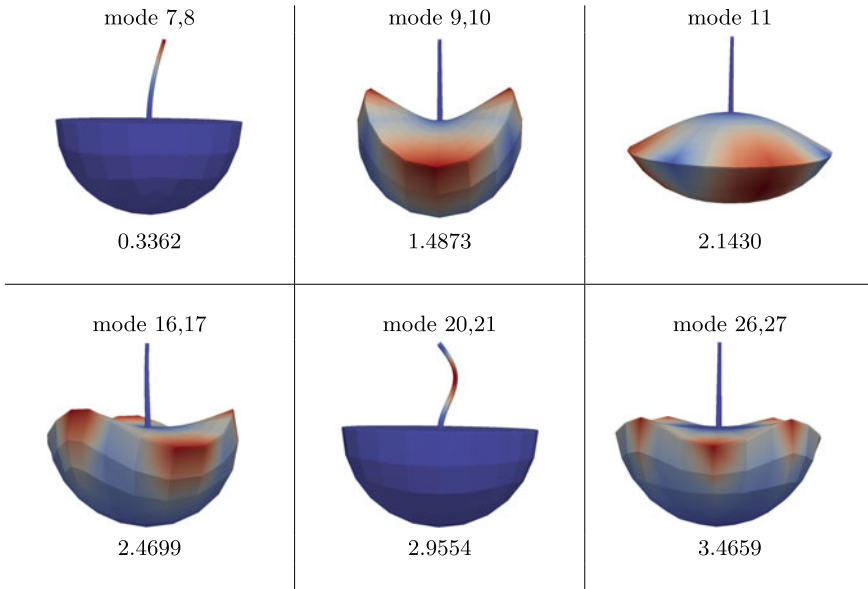
The towers first three bending modes are represented by mode (1, 2) (3, 4) and (6, 7), since the system is symmetric they all appear twice one in x- and one in y-direction. Mode 5 is a stretch mode. The more complex continuum model lead to very similar natural frequencies, whereby the number of the modes differ.

13.3.1.2 Tower with Foundation and Soil

Foundation and soil are modeled by three dimensional continuum elements. The finite element mesh is fine in the center and becomes more coarse to the outside. So that the complete model consists of 14052 finite elements with 18140 nodes and 54420 DOF. The cylindrical reinforced concrete foundation has the material parameters: Young's modulus $E = 4.896 \cdot 10^{10}$ N/m², Poisson's ratio $\nu = 0.2$ and density $\rho = 3054$ kg/m³. Material parameters of concrete and steel are smeared, as discussed before. The surrounding soil is modeled as a homogeneous isotropic half-sphere with a radius of 102 m. It consists of close sand with the material parameters: Young's modulus $E = 1.5 \cdot 10^8$ N/m², Poisson's ratio $\nu = 0.25$ and density $\rho = 2200$ kg/m³. The wind turbine's tower is mounted to the foundation directly. Here no fixed-base boundary conditions are applied.

Solving the eigenvalue problem leads to the bending modes of the tower, but also includes eigenmodes of the foundation and the soil (cf. Table 13.3) as well as their

Table 13.3 Natural frequencies f [Hz] of the towers model including foundation and surrounding soil



combinations. The first and second bending mode of the tower are shifted slightly, due to the influence of the soil flexibility.

13.3.2 Scaled Boundary Finite Element Model

The infinite half-space is discretized by scaled boundary finite elements. The nodes and elements are located at the common interface for the finite element model and the scaled boundary finite element model. Since only the surface is discretized 132 elements with 145 nodes, with 3 DOF each, are needed. The scaling center is located in the domains center directly beyond the towers bottom with the global coordinates $SC = \{0, 0, 0\}$.

13.3.3 Load

The estimation of the loading functions $p_{\Omega\Omega}$ and $p_{\Gamma\Gamma}$ is a crucial aspect of the analysis strategy, as the frequency content and the length of the excitation signals influence the performance of the method. Moreover, if a seismic base excitation need

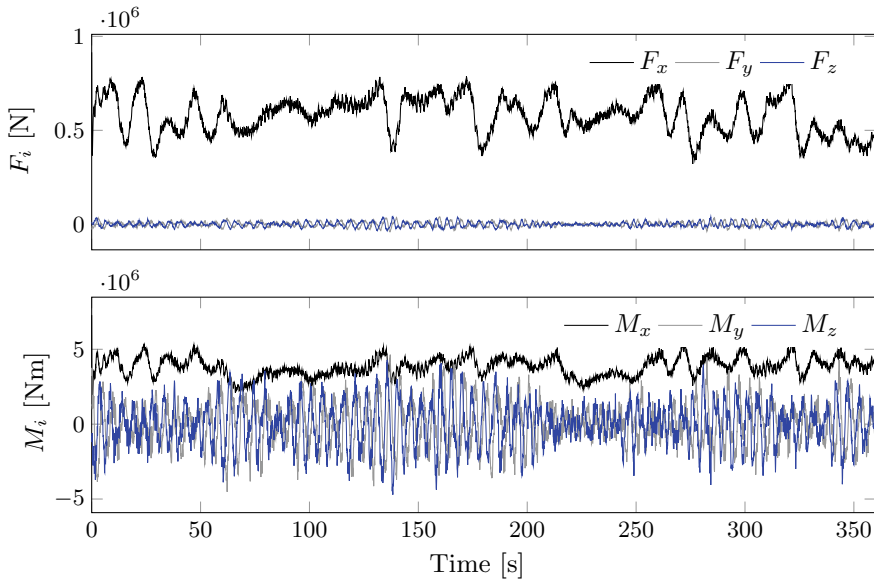


Fig. 13.3 Aerodynamic loads

to be accounted for, it is necessary to perform a pre-step for the estimation of the free field loads at the interaction nodes. The estimation of the aerodynamic and seismic loading functions is carried out according to [15]. It is assumed that the aerodynamic loads are not noticeably influenced by the SSI and can be computed for a fixed base tower.

The aerodynamic loads are computed by utilizing FAST [7], they can be applied to the beam model directly. For the computation of the turbulent aerodynamic loads, a reference speed of 12 m/s was used. The turbulent wind field model used corresponds to the spectral and exponential coherence model of Kaimal, which also meets the requirements of the standards [5]. It is assumed the aerodynamic loads are not noticeably influenced by the SSI and can be computed for a fixed base tower. The loads F_x and F_y act in fore-aft direction and side-side direction, respectively. F_z acts in axial direction. The moments M_x , M_y , and M_z are roll, pitch and yaw moments, respectively. In case of the continuum model the loads have to be distributed to the top nodes of the system, which comprises a higher effort in modeling especially for the aerodynamic moments (Fig. 13.3).

A seismic event accelerates the structure and lead to additional excitations. Here a strong earthquake event has been chosen to conduct the numerical analysis. Figure 13.4 shows the acceleration-time-plot of the Kobe, Japan earthquake in 1995. It has a moment magnitude scale of $M_w = 7.2$.

Since accelerations can not be applied directly to the equation of motion (cf. Eq. (13.6)), accept of initial condition, equivalent earthquake loads are needed. They can be computed with

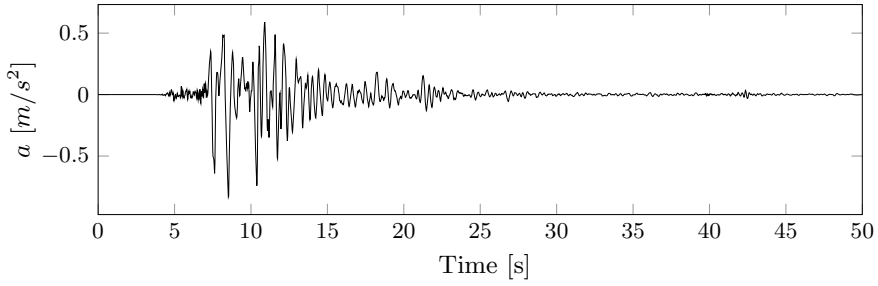


Fig. 13.4 Earthquake acceleration Kobe, Japan, 1995

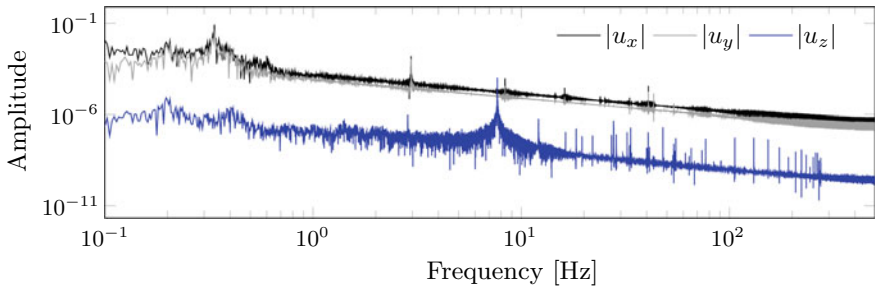


Fig. 13.5 Response of the undamped system

$$p_{eq}(t) = M\mathbf{1}a(t), \tag{13.7}$$

where $p_{eq}(t)$ denotes the time-dependent earthquake load. M is the mass matrix, as defined before. $\mathbf{1}$ is a vector that contains '1' at the degrees of freedom the earthquake load is addressed to. $a(t)$ contains the scalar information of acceleration.

13.3.4 Damping

In order to simulate a realistic behavior of the wind turbine damping is introduced. Since the damping parameters of the real structure are unknown and difficult to determine the load is applied to the undamped system to figure out which eigenmodes are addressed, see Fig. 13.5.

The presented model considers Rayleigh damping so that the coefficients c_M and c_K are needed to build the damping matrix (cf. Eq. (13.4)). The damping ratio is assumed to be $\approx 2\%$ in case of the steel tower and the geometrical damping of the soil leads to a $\approx 5\%$ damping ratio at higher frequencies, in particular at those higher than the towers second eigenfrequency. It is possible to compute c_M and c_K by evaluating the damping coefficient

$$\delta_i = \frac{c_M}{4\pi f_i} + c_K \pi f_i. \quad (13.8)$$

The terms $\frac{c_M}{4\pi f_i}$ and $c_K \pi f_i$ prescribe mass proportional damping and stiffness proportional damping, respectively. An eigenvalue analysis yields to the eigenfrequencies f_i each eigenfrequency corresponds to one damping coefficient δ_i . Choosing two eigenfrequencies and two corresponding damping coefficients c_M and c_K are defined, as well as all other damping coefficients for all other eigenfrequencies. Here $f_7 = 0.3362$ Hz, $\delta_7 = 2\%$ and $f_{26} = 3.4659$ Hz, $\delta_{26} = 5\%$ are chosen, they represent the first towers bending mode and an arbitrary eigenmode of the soil, which is bigger than the towers second mode (cf. Table 13.3), respectively. This yield to Rayleigh damping coefficients $c_M = 0.06183$ and $c_K = 0.00507$ so that damping matrix D can be assembled.

13.4 Case Study

The case study is conducted for both models with a time step length of $\Delta t = 0.01$ [s] for a total period of $T = 360$ [s]. As mentioned the time stepping is done by utilizing the generalized- α scheme with a parameter $\rho^\infty = 0.8$. The aerodynamic loads (cf. Fig. 13.3) are applied as described at the towers top.

13.4.1 Tower Under Aerodynamic Load

Figure 13.6 depicts the results of the towers top and the base displacement of the two numerical models. For each case, the displacements in the x-, y- and z-direction were shown. When looking at the results, it becomes clear that the calculation, taking into account soil and foundations, leads to different results, then the simple cantilever. At the tower top there are different amplitudes observed and additionally phase shifts is present, due to the change of eigenfrequencies (cf. Table 13.2). In case of the full model considering tower, foundation and half space displacements of the towers base can be observed. Since this base displacements are very small, compared to the top, the results of the top displacements of the two models are very similar. In case of a smaller foundations or softer surrounding soil the base displacements will increase and so the top displacement.

13.4.2 Tower Under Aerodynamic and Seismic Load

The setup is the same as before, but at $t = 120$ [s] seismic load is applied to the structure. This results as expected into an increase of the towers top displacement. After the seismic event is subsided, the towers motion is comparable to the previous simulation without seismic event (Fig. 13.7).

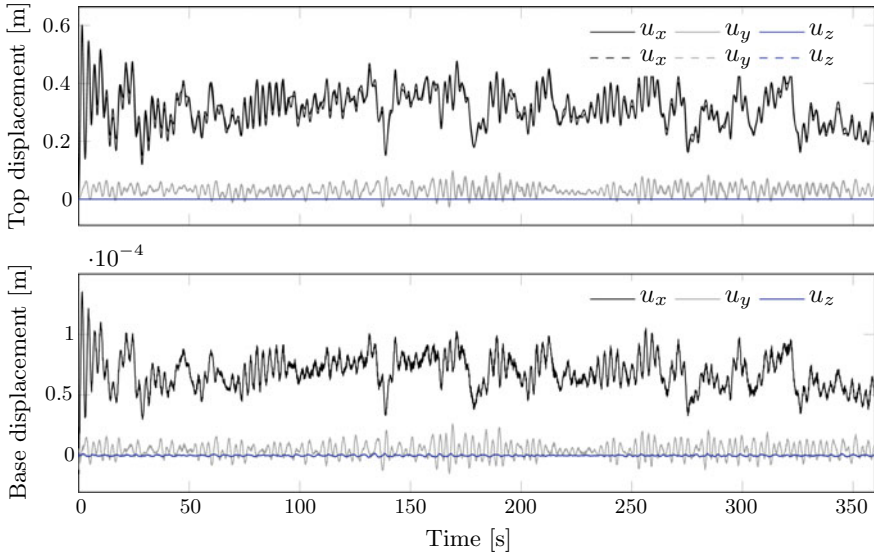


Fig. 13.6 Top and base displacement of the two models under aerodynamic load. Solid line represents the tower with foundation and infinite half-space and the dashed line represents the tower as a cantilever

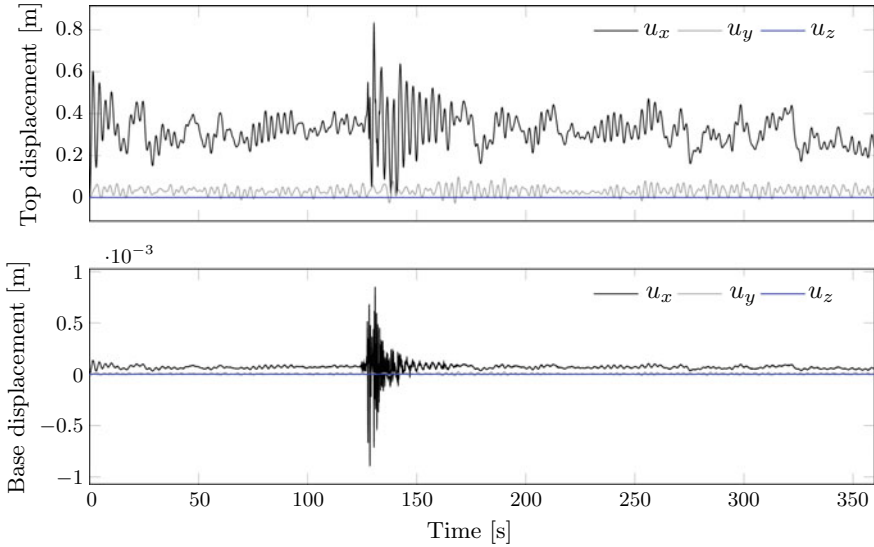


Fig. 13.7 Top and base displacement of the tower with foundation and infinite half-space under aerodynamic and seismic load

13.5 Conclusion and Outlook

In this article we presented a strategy to simulate operating wind turbine, considering the soil-structure-interaction, with the aid of a coupled FEM-SBFEM approach in time domain. The proposed method is able to represent the salient aspects of the SSI interaction and allows arbitrary transient loading conditions, such as aerodynamic loads, seismic loads and the combinations of them.

In future works, nonlinear effects shall be taken into account, thanks to the suitability of this strategy for transient time-domain analysis. The nonlinearities can be accounted for in the structure, in the near field and at the soil-structure interface. Concerning the soil (near field) nonlinearity, most of the models used in practice consider a simple one-dimensional behavior. The proposed method is able to investigate the 3D nonlinear response of the near field, as the nonlinearity can be assigned to any element of the model. However, a linear behavior of the far-field soil is assumed.

Moreover, the seismic input can be expressed as a 3D seismic wave field and transformed into boundary tractions, which are then applied at the interface between the near and far fields [2]. This would improved the description of the scattered incoming seismic wave field and lead to more realistic simulations.

Finally, more sophisticated tower models, with openings or non-axial-symmetrical shapes, can be straightforwardly investigated.

References

1. Balay, S., et al.: PETSc Users Manual. Argonne National Laboratory, ANL-95/11 - Revision 3.6, <http://www.mcs.anl.gov/petsc> (2015)
2. Bazyar, M.H., et al.: Analysis of transient wave scattering and its applications to site response analysis using the scaled boundary finite-element method. *Soil Dyn. Earthq. Eng.* **98**, 191–205 (2017)
3. Blackford, L.S., et al.: ScaLAPACK Users Guide. Society for Industrial and Applied Mathematics, Philadelphia (1997)
4. Chung, J., et al.: A time integration algorithm for structural dynamics with improved numerical dissipation: the generalized- α method. *J. Appl. Mech.* **60**, 372–375 (1993)
5. DIN EN 61400-1:2011-08: Windenergieanlagen - Teil 1: Auslegungsanforderungen, in German (2011)
6. GNU Octave Scientific Programming Language. <https://www.gnu.org/software/octave/>
7. Jonkman, J., et al.: FAST user's guide. Technical Report, NREL/TP-500-38230 (2005)
8. Jonkman, J., et al.: Definition of a 5-MW reference wind turbine for offshore system development. Technical Report, NREL/TP-500-38060 (2009)
9. Kitware: ParaView 5.0.1. <https://www.paraview.org/>
10. NREL National Renewable Energy Laboratory: FAST v7. <https://nwtc.nrel.gov/FAST>
11. Sandia National Laboratories: New Mexico, CUBIT 13.2. <https://cubit.sandia.gov/index.html>
12. Schauer, M., Langer, S.: Implementation of an efficient coupled FEM-SBFEM approach for soil-structure-interaction analysis. In: Schrefler, B.A., Oñate, E., Papadrakakis, M. (eds.) Proceedings of the VI International Conference on Coupled Problems in Science and Engineering, pp. 370–381 (2015)

13. Schauer, M., Langer, S., Roman, E.J., Quintana-Ortí, E.S.: Large scale simulation of wave propagation in soils interacting with structures using FEM and SBFEM. *J. Comput. Acoust.* **19**(1), 75–93 (2011)
14. Schauer, M., Roman, E.J., Quintana-Ortí, E.S., Langer, S.: Parallel computation of 3-D soil-structure interaction in time domain with a coupled FEM/SBFEM approach. *J. Sci. Comput.* **52**, 446–467 (2012)
15. Taddei, F.: Numerical investigation of soil-structure interaction for onshore wind turbines grounded on a layered soil. Ph.D. thesis, RWTH Aachen University (2014)
16. Wolf, J.: *The Scaled Boundary Finite Element Method*. Wiley, Chichester (2003)

Chapter 14

Constraint Coupling for Flexible Multibody Systems: Stabilization by Modified Spatial Discretization



Fabio Schneider and Michael Burger

Abstract We consider a differential-algebraic co-simulation approach to couple flexible structures to a rigid multibody system. That is, the coupling is realized using an algebraic constraint equation. The spatial discretization of flexible structures introduces an algebraic loop in the data exchange of subsystems. We investigate how this influences the stability and how modifying the discretization can help to stabilize the co-simulation.

14.1 Introduction

In mechanical or mechatronical system modeling, many different subsystems and their interaction have to be considered. Consequently, hybrid models which combine, e.g., electrical circuits, fluid dynamics, flexible components and rigid multibody models (see e.g. [2, 3, 15]) have to be coupled and simulated.

In general, the mathematical models of these systems strongly differ, for instance in the formulation, the complexity or the time constants. Thus, simulation of the complete system is a challenging task. However, co-simulation allows to solve all subsystems with specially suited methods, while the interaction is handled via data exchange at discrete communication points in time, the so-called macro time points (cf. [4–7]).

Recently, we developed a force-displacement co-simulation strategy for kinematic coupling of mechanical systems and in particular flexible multibody systems [12], [13]. It is based on constraint coupling, i.e. algebraic constraints describe the kinematic coupling. The stability analysis shows, that subsystem masses are decisive for stable co-simulation. Now, we consider how modified spatial discretization can be used to influence and to improve the stability of our co-simulation approach.

F. Schneider (✉) · M. Burger

Fraunhofer Institute for Industrial Mathematics, Fraunhofer-Platz 1, 67663

Kaiserslautern, Germany

e-mail: fabio.schneider@itwm.fraunhofer.de

© Springer Nature Switzerland AG 2019

B. Schweizer (ed.), *IUTAM Symposium on Solver-Coupling and Co-Simulation*,

IUTAM Bookseries 35, https://doi.org/10.1007/978-3-030-14883-6_14

In particular, the paper at hand is organized as follows. First, we briefly present our differential-algebraic coupling approach and the corresponding stability analysis. A linear 2-mass spring-damper test model is used to illustrate the method. After that, we consider the stability of the co-simulation, depending on the spatial discretization. To this end, we regard both equidistant and non-equidistant spatial discretization. Finally, the coupling approach is also applied on a complex nonlinear application example.

14.2 Co-simulation Using a Kinematic Coupling Constraint

The formulation of a kinematic coupling in flexible multibody systems is very problem-specific. Often, it is designed especially for the current application and assumed simplifications are not valid in general. Moreover, a common strategy is the introduction of a coupling stiffness, which is mostly unknown and, thus, it is an artificial parameter. Naturally, this additional parameter can lead to stiff systems.

In contrast, we have developed a general coupling strategy from a differential-algebraic approach [12, 13]. It does not introduce additional parameters and does not assume a priori simplifications. Moreover, it allows to easily describe complex coupling joints via the formulation of algebraic constraints.

14.2.1 The Differential-Algebraic Coupling Approach

Let the rigid multibody system and the flexible structure with absolute coordinates $\mathbf{q}_R \in \mathbb{R}^{n_R}$, $\mathbf{q}_F \in \mathbb{R}^{n_F}$ and corresponding velocities $\mathbf{v}_R \in \mathbb{R}^{n_R}$, $\mathbf{v}_F \in \mathbb{R}^{n_F}$ be given as ordinary differential equations of motion

$$\mathbf{M}_R \dot{\mathbf{v}}_R = \mathbf{f}_R(\mathbf{q}_R, \mathbf{v}_R) \quad \text{and} \quad \mathbf{M}_F \dot{\mathbf{v}}_F = \mathbf{f}_F(\mathbf{q}_F, \mathbf{v}_F), \quad (14.1)$$

where the mass matrices \mathbf{M}_R and \mathbf{M}_F may depend on the states \mathbf{q}_R , resp. \mathbf{q}_F .

To keep the notation simple, we assume $\dot{\mathbf{q}}_R = \mathbf{v}_R$ and $\dot{\mathbf{q}}_F = \mathbf{v}_F$ for the kinematic differential equations.

To derive the coupling equations, we formulate the corresponding kinematic coupling constraints of the subsystems

$$\mathbf{0} = \mathbf{g}_{co}(\mathbf{q}_R^{co}, \mathbf{q}_F^{co}) \in \mathbb{R}^{n_{co}}, \quad (14.2)$$

where the n_{co} constraints are assumed to be independent and only a part of all the states in \mathbf{q}_R and \mathbf{q}_F are involved in the coupling. We define them as $\mathbf{q}_R^{co} \in \mathbb{R}^{n_{R,co}}$ and $\mathbf{q}_F^{co} \in \mathbb{R}^{n_{F,co}}$. The remaining inner states $\mathbf{q}_R^{in} \in \mathbb{R}^{n_{R,in}}$ and $\mathbf{q}_F^{in} \in \mathbb{R}^{n_{F,in}}$ do not appear in the coupling constraints (cf. Fig. 14.1). With this partitioning, the coupling constraint (14.2) and Lagrange multipliers $\boldsymbol{\Lambda} \in \mathbb{R}^{n_{co}}$, the subsystem Eq. (14.1) lead to the differential-algebraic system

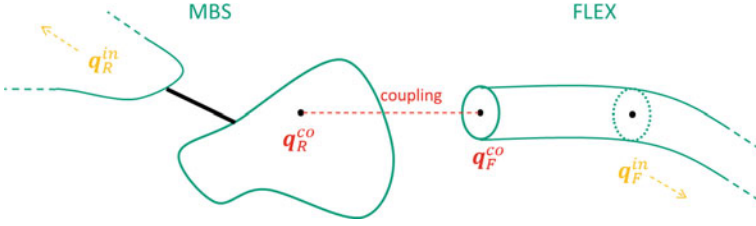


Fig. 14.1 Coupling of a multibody system and a flexible structure (e.g. a cable) with coupling states \mathbf{q}_R^{co} and \mathbf{q}_F^{co}

$$\begin{pmatrix} \mathbf{M}_R^{co} & \mathbf{M}_R^{ic} \\ \mathbf{M}_R^{ci} & \mathbf{M}_R^{in} \end{pmatrix} \begin{bmatrix} \dot{\mathbf{v}}_R^{co} \\ \dot{\mathbf{v}}_R^{in} \end{bmatrix} = \begin{bmatrix} \mathbf{f}_R^{co}(\mathbf{q}_R, \mathbf{v}_R) \\ \mathbf{f}_R^{in}(\mathbf{q}_R, \mathbf{v}_R) \end{bmatrix} - \begin{pmatrix} \frac{\partial \mathbf{g}_{co}}{\partial \mathbf{q}_R}{}^T \\ \mathbf{0} \end{pmatrix} \Lambda, \quad (14.3a)$$

$$\begin{pmatrix} \mathbf{M}_F^{co} & \mathbf{M}_F^{ic} \\ \mathbf{M}_F^{ci} & \mathbf{M}_F^{in} \end{pmatrix} \begin{bmatrix} \dot{\mathbf{v}}_F^{co} \\ \dot{\mathbf{v}}_F^{in} \end{bmatrix} = \begin{bmatrix} \mathbf{f}_F^{co}(\mathbf{q}_F, \mathbf{v}_F) \\ \mathbf{f}_F^{in}(\mathbf{q}_F, \mathbf{v}_F) \end{bmatrix} - \begin{pmatrix} \frac{\partial \mathbf{g}_{co}}{\partial \mathbf{q}_F}{}^T \\ \mathbf{0} \end{pmatrix} \Lambda, \quad (14.3b)$$

$$\mathbf{0} = \mathbf{g}_{co}(\mathbf{q}_R^{co}, \mathbf{q}_F^{co}). \quad (14.3c)$$

From that, we want to derive a force-displacement coupling, where prescribed coupling states are given to the flexible structure and the multibody system receives coupling forces as feedback. Consequently, the flexible structure can be interpreted as a force element. For more details on multibody dynamics we refer to [11, 17].

Assuming $\frac{\partial \mathbf{g}_{co}}{\partial \mathbf{q}_F^{co}}$ is quadratic (i.e. $\mathbf{q}_F^{co} \in \mathbb{R}^{n_{co}}$ and thus $\frac{\partial \mathbf{g}_{co}}{\partial \mathbf{q}_F^{co}} \in \mathbb{R}^{n_{co} \times n_{co}}$) and non-singular, we can apply the implicit function theorem to the constraint Eq. (14.2) and its derivatives w.r.t. time. For the hidden constraints, i.e. the constraint equations on velocity and acceleration level, the linearity allows an explicit solution for the coupling states of the flexible structure

$$\begin{aligned} 0 &= \frac{\partial \mathbf{g}_{co}}{\partial \mathbf{q}_{Rco}} \mathbf{v}_R^{co} + \frac{\partial \mathbf{g}_{co}}{\partial \mathbf{q}_F^{co}} \mathbf{v}_F^{co} \implies \mathbf{v}_F^{co} = - \left(\frac{\partial \mathbf{g}_{co}}{\partial \mathbf{q}_F^{co}} \right)^{-1} \frac{\partial \mathbf{g}_{co}}{\partial \mathbf{q}_{Rco}} \mathbf{v}_R^{co}, \\ 0 &= \frac{\partial \mathbf{g}_{co}}{\partial \mathbf{q}_{Rco}} \dot{\mathbf{v}}_R^{co} + \frac{\partial \mathbf{g}_{co}}{\partial \mathbf{q}_F^{co}} \dot{\mathbf{v}}_F^{co} + \mathbf{g}_{co}'' \implies \dot{\mathbf{v}}_F^{co} = - \left(\frac{\partial \mathbf{g}_{co}}{\partial \mathbf{q}_F^{co}} \right)^{-1} \left(\frac{\partial \mathbf{g}_{co}}{\partial \mathbf{q}_{Rco}} \dot{\mathbf{v}}_R^{co} + \mathbf{g}_{co}'' \right). \end{aligned} \quad (14.4)$$

On position level, in general only an implicit expression for \mathbf{q}_F^{co} exists and would require an iterative solution. In practice, however, often also \mathbf{q}_F^{co} can be computed explicitly.

In total, this allows to solve for the coupling states of the flexible structure \mathbf{q}_F^{co} , \mathbf{v}_F^{co} and $\dot{\mathbf{v}}_F^{co}$, which then serve as displacement inputs.

Consequently, in (14.3b) we can skip the corresponding equations of motion of the coupling state \mathbf{q}_F^{co} and only proceed with the lower block line

$$\mathbf{M}_F^{in} \dot{\mathbf{v}}_F^{in} = \mathbf{f}_F^{in}(\mathbf{q}_F^{co}, \mathbf{q}_F^{in}, \mathbf{v}_F^{co}, \mathbf{v}_F^{in}) - \mathbf{M}_F^{ci} \dot{\mathbf{v}}_F^{co}, \quad (14.5)$$

such that the constraint forces acting on the flexible structure are eliminated.

For the multibody system we have to compute constraint forces, or more precisely, the Lagrange multiplier Λ . It can be computed from the above skipped upper block line of (14.3b) to get (as already proposed in [16])

$$\Lambda = \left(\frac{\partial \mathbf{g}_{co}}{\partial \mathbf{q}_F^{co}} \right)^{-1} \left(\mathbf{f}_F^{co}(\mathbf{q}_F^{co}, \mathbf{q}_F^{in}, \mathbf{v}_F^{co}, \mathbf{v}_F^{in}) - \mathbf{M}_F^{co} \dot{\mathbf{v}}_F^{co} - \mathbf{M}_F^{ic} \dot{\mathbf{v}}_F^{in} \right). \tag{14.6}$$

Summarizing, we end up with a force-displacement coupling, where displacements and constraint forces are exchanged in a co-simulation.

14.2.2 Co-Simulation

The co-simulation proceeds in discrete macro time steps $t_k \rightarrow t_{k+1} = t_k + h$ with macro time step size h , where inside of each macro time step the subsystems integrate independently of each other. Only at the macro time points data is exchanged. As described above, inputs of the flexible structure are defined as $\mathbf{u}_F = [\mathbf{q}_F^{coT}, \mathbf{v}_F^{coT}, \dot{\mathbf{v}}_F^{coT}]^T$, whereas for the multibody system we have $\mathbf{u}_R = \Lambda$. Corresponding outputs \mathbf{y}_R and \mathbf{y}_F are defined such that it holds $\mathbf{u}_F = \mathbf{y}_R$ and $\mathbf{u}_R = \mathbf{y}_F$.

To integrate up to the next macro time point t_{k+1} , also inputs at intermediate points in time, i.e. in $[t_k, t_{k+1}]$, are necessary and must be approximated.

Figure 14.2 shows the scheme for a parallel co-simulation (Jacobi type), where both subsystems integrate up to the next macro time point in parallel. Thus, input data must be predicted from previous macro time points. To this end, extrapolation of order r is used, i.e. input data from previous macro time points $\mathbf{u}_k = \mathbf{u}(t_k), \dots, \mathbf{u}_{k-r} = \mathbf{u}(t_{k-r})$ is extrapolated to the interval $[t_k, t_{k+1}]$. This is done by evaluating the interpolation polynomial

$$\mathcal{T}_m^r(t, t_k, \dots, t_{k-r}) \begin{bmatrix} \mathbf{u}_k \\ \vdots \\ \mathbf{u}_{k-r} \end{bmatrix} \tag{14.7}$$

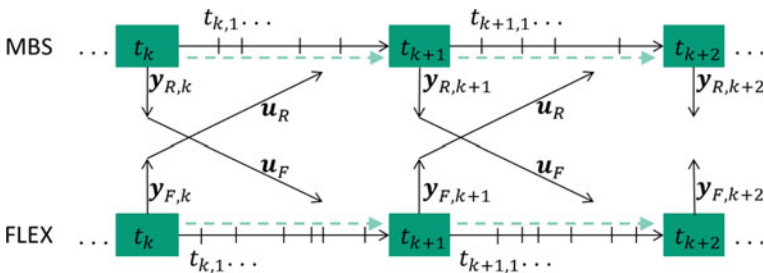


Fig. 14.2 Parallel co-simulation (Jacobi type) with extrapolated inputs

at time $t = t_k + \tau h$ with $0 \leq \tau \leq 1$. The lower index m in \mathcal{T}_m^r indicates the dimension of the extrapolated quantity, i.e. here $\mathbf{u}_k \in \mathbb{R}^m$.

For a fixed macro time step size h and extrapolation of order r at $t = t_k + \tau h$, we simply write $\mathcal{T}_m^r(\tau)$ for the extrapolation operator. With constant ($r = 0$), linear ($r = 1$) and quadratic ($r = 2$) extrapolation, it is given as

$$\begin{aligned} \mathcal{T}_m^0(\tau) &= \mathbb{I}_m, & \mathcal{T}_m^1(\tau) &= ((1 + \tau)\mathbb{I}_m \ (-\tau)\mathbb{I}_m), \\ \mathcal{T}_m^2(\tau) &= \left(\left(1 + \frac{3\tau}{2} + \frac{\tau^2}{2} \right) \mathbb{I}_m \ (-2\tau - \tau^2)\mathbb{I}_m \ \left(\frac{\tau}{2} + \frac{\tau^2}{2} \right) \mathbb{I}_m \right). \end{aligned} \quad (14.8)$$

14.3 Stability of the Coupling Approach

In this section, we investigate the stability of the presented coupling approach. To this end, we start with theoretical considerations and then apply the stability analysis on a test problem.

14.3.1 Stability Analysis

We formulate the iterative behavior of subsequent macro time steps $t_k \rightarrow t_{k+1} = t_k + h$ with macro time step size h . For this purpose, we consider two first order linear subsystems in state space form

$$\begin{aligned} \dot{\mathbf{x}}_R &= \mathbf{A}_R \mathbf{x}_R + \mathbf{B}_R \mathbf{u}_R & \text{and} & & \dot{\mathbf{x}}_F &= \mathbf{A}_F \mathbf{x}_F + \mathbf{B}_F \mathbf{u}_F \\ \mathbf{y}_R &= \mathbf{C}_R \mathbf{x}_R + \mathbf{D}_R \mathbf{u}_R & & & \mathbf{y}_F &= \mathbf{C}_F \mathbf{x}_F + \mathbf{D}_F \mathbf{u}_F \end{aligned} \quad (14.9)$$

which can be understood as the linearization around equilibrium points of subsystems (14.3a) and (14.5), together with linearized output equations. Moreover, it holds $\mathbf{u}_R = \mathbf{y}_F$ and $\mathbf{u}_F = \mathbf{y}_R$.

With compact notations for the system states, outputs and inputs

$$\mathbf{x} := \begin{bmatrix} \mathbf{x}_R \\ \mathbf{x}_F \end{bmatrix}, \quad \mathbf{y} := \begin{bmatrix} \mathbf{y}_R \\ \mathbf{y}_F \end{bmatrix}, \quad \mathbf{u} := \begin{bmatrix} \mathbf{u}_R \\ \mathbf{u}_F \end{bmatrix},$$

the matrices of the state space systems

$$\mathbf{A} := \begin{pmatrix} \mathbf{A}_R & \\ & \mathbf{A}_F \end{pmatrix}, \quad \mathbf{B} := \begin{pmatrix} \mathbf{B}_R & \\ & \mathbf{B}_F \end{pmatrix}, \quad \mathbf{C} := \begin{pmatrix} \mathbf{C}_R & \\ & \mathbf{C}_F \end{pmatrix}, \quad \mathbf{D} := \begin{pmatrix} \mathbf{D}_R & \\ & \mathbf{D}_F \end{pmatrix},$$

the data exchange $\mathbf{u} = \mathbf{L}\mathbf{y}$ with $\mathbf{L} = \begin{pmatrix} 0 & \mathbb{I} \\ \mathbb{I} & 0 \end{pmatrix}$ and the extrapolation

$$\mathbf{u}(t_k + \tau h) \approx \mathbf{T}_{n_u}^r(\tau) \begin{bmatrix} \mathbf{u}_k \\ \vdots \\ \mathbf{u}_{k-r} \end{bmatrix} \tag{14.10}$$

of order r , we perform an exact time integration on $[t_k, t_{k+1}]$, which results in the propagation from t_k to t_{k+1}

$$\begin{bmatrix} \mathbf{x}_{k+1} \\ \vdots \\ \mathbf{x}_{k+1-r} \\ \mathbf{y}_{k+1} \\ \vdots \\ \mathbf{y}_{k+1-r} \end{bmatrix} = \underbrace{\begin{pmatrix} \mathcal{S}_{xx}(h) & \mathcal{S}_{xy}(h) \\ \mathbb{I}_{n_x} \mathbf{0} & \mathbf{0} \dots \mathbf{0} \\ \vdots & \vdots \\ \mathcal{S}_{yx}(h) & \mathcal{S}_{yy}(h) \\ \mathbf{0} \dots \mathbf{0} & \mathbb{I}_{n_y} \mathbf{0} \\ \vdots & \vdots \\ \mathbf{0} \dots \mathbf{0} & \vdots \\ & \mathbb{I}_{n_y} \mathbf{0} \end{pmatrix}}_{=:\mathcal{S}(h)} \begin{bmatrix} \mathbf{x}_k \\ \vdots \\ \mathbf{x}_{k-r} \\ \mathbf{y}_k \\ \vdots \\ \mathbf{y}_{k-r} \end{bmatrix} \tag{14.11}$$

with matrices

$$\mathcal{S}_{xx}(h) = (e^{Ah} \mathbf{0} \dots \mathbf{0}) + h \int_0^1 e^{A(h-\tau h)} \mathbf{BLCT}_{n_x}^r(\tau) d\tau, \tag{14.12a}$$

$$\mathcal{S}_{xy}(h) = h \int_0^1 e^{A(h-\tau h)} \mathbf{BLDLT}_{n_y}^r(\tau) d\tau, \tag{14.12b}$$

$$\begin{aligned} \mathcal{S}_{yx}(h) &= (\mathbf{C}e^{Ah} \mathbf{0} \dots \mathbf{0}) + Ch \int_0^1 e^{A(h-\tau h)} \mathbf{BLCT}_{n_x}^r(\tau) d\tau \\ &= \mathbf{CS}_{xx}(h), \end{aligned} \tag{14.12c}$$

$$\begin{aligned} \mathcal{S}_{yy}(h) &= Ch \int_0^1 e^{A(h-\tau h)} \mathbf{BLDLT}_{n_y}^r(\tau) d\tau + \mathbf{DLT}_{n_y}^r(1) \\ &= \mathbf{CS}_{xy}(h) + \mathbf{DLT}_{n_y}^r(1). \end{aligned} \tag{14.12d}$$

In long-term simulation, a stable co-simulation can only be guaranteed, if the spectral radius of $\mathcal{S}(h)$ fulfills

$$\rho(\mathcal{S}(h)) < 1. \tag{14.13}$$

Besides the stability for finite time step sizes h , zero-stability (i.e. the stability for $h \rightarrow 0$) often is considered. In this case, Eq. (14.11) simplifies to $\mathbf{x}_{k+1} = \mathbf{x}_k$, i.e. \mathbf{x} is constant, and

$$\begin{bmatrix} \mathbf{y}_{k+1} \\ \vdots \\ \mathbf{y}_{k+1-r} \end{bmatrix} = \begin{bmatrix} \mathbf{C}\mathbf{x}_k \\ \mathbf{0} \\ \vdots \\ \mathbf{0} \end{bmatrix} + \underbrace{\begin{pmatrix} \mathbf{DLT}_{n_y}^r(1) & \\ \mathbb{I}_{n_y} & \mathbf{0} \\ \vdots & \vdots \\ \mathbb{I}_{n_y} & \mathbf{0} \end{pmatrix}}_{=:\mathcal{S}_0} \begin{bmatrix} \mathbf{y}_k \\ \vdots \\ \mathbf{y}_{k-r} \end{bmatrix}. \tag{14.14}$$

For zero-stability, the spectral radius of \mathcal{S}_0 must fulfill $\rho(\mathcal{S}_0) < 1$. The matrix \mathcal{S}_0 only depends on \mathbf{DL} and the order of extrapolation, since in $\mathcal{T}_{n_y}^r(1)$, i.e. for $\tau = 1$, only the constant coefficients remain.

If either \mathbf{D}_R or \mathbf{D}_F vanishes, i.e. \mathbf{y}_R does not explicitly depend on $\mathbf{u}_R = \mathbf{y}_F$ or vice versa (see the state space Eq. (14.9)), no algebraic loop exists and it holds $\rho(\mathcal{S}_0) = 0$. In this case, zero-stability is given. In [9] this is investigated for constant extrapolation ($r = 0$).

Moreover, looking at the characteristic polynomial of \mathcal{S}_0 in more detail, one can find a contractivity condition on subsystem mass ratios, which we will illustrate in the next section. A similar result can be found in [1] for two coupled multibody systems.

14.3.2 Stability of the Test Problem: 2-Mass Spring-Damper Model

To test the presented co-simulation method, we set up a simple model for numerical experiments, similar to the one used in [5]. The multibody system only has one degree of freedom \mathbf{q}_R , while the flexible structure consists of two degrees of freedom \mathbf{q}_F^{co} and \mathbf{q}_F^{in} . The system parameters are given in Table 14.1, where the total mass of the flexible structure $m_F^{co} + m_F^{in}$ will be varied for the stability analysis.

The kinematic coupling is formulated with the algebraic constraint $0 = \mathbf{q}_R - \mathbf{q}_F^{co}$ and results in a 2-mass spring-damper model, as shown in Fig. 14.3.

Applying the presented force-displacement coupling approach, this leads to

$$m_R \dot{\mathbf{v}}_R = -c_R \mathbf{q}_R - d_R \mathbf{v}_R - \mathbf{u}_R \tag{14.15a}$$

$$\mathbf{y}_R = [\mathbf{q}_F^{co}, \mathbf{v}_F^{co}, \dot{\mathbf{v}}_F^{co}]^T = [\mathbf{q}_R, \mathbf{v}_R, \dot{\mathbf{v}}_R]^T \tag{14.15b}$$

Table 14.1 System parameters for the basic 2-mass spring-damper test problem. The mass $m_F^{co} + m_F^{in}$ will be varied for the stability analysis

Parameter	m_R	m_F^{co}	m_F^{in}	c_R	d_R	c_F	d_F
Value	5 kg	0.5 kg	0.5 kg	100 N/m	1 Ns/m	50 N/m	0.1 Ns/m

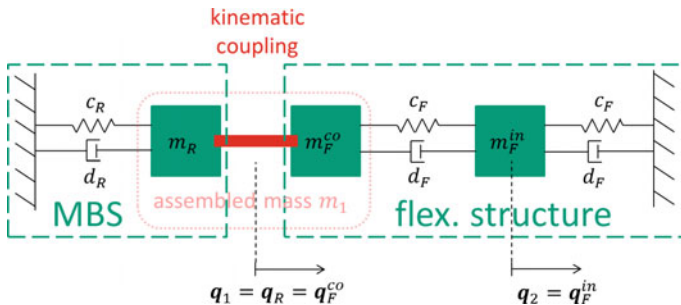


Fig. 14.3 Test model: 2-mass spring-damper model with kinematic coupling

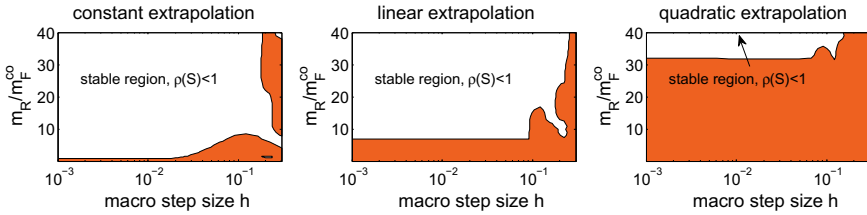


Fig. 14.4 Stability regions depending on the mass ratio and the macro time step size h with constant ($r = 0$), linear ($r = 1$) and quadratic ($r = 2$) extrapolation

for the multibody system and

$$m_F^{in} \dot{\mathbf{v}}_F^{in} = -2c_F \mathbf{q}_F^{in} - 2d_F \mathbf{v}_F^{in} + c_F \mathbf{u}_F^{(1)} + d_F \mathbf{u}_F^{(2)} \tag{14.16a}$$

$$\mathbf{y}_F = \mathbf{A} = -c_F \mathbf{q}_F^{in} - d_F \mathbf{v}_F^{in} + c_F \mathbf{u}_F^{(1)} + d_F \mathbf{u}_F^{(2)} + m_F^{co} \mathbf{u}_F^{(3)} \tag{14.16b}$$

for the flexible structure. In both output equations, the explicit dependence on inputs is obvious, such that an algebraic loop exists. As before, for the inputs it holds $\mathbf{u}_F = \mathbf{y}_R$ and $\mathbf{u}_R = \mathbf{y}_F$.

If we apply the stability analysis on the test problem and plot $\rho(\mathcal{S}(h))$ depending on h and the mass ratio $\frac{m_R}{m_F^{co}}$, we get the results shown in Fig. 14.4. Here, the stable areas with $\rho(\mathcal{S}(h)) < 1$ are plotted in white and the unstable regions are plotted in red.

We want to emphasize two observations. First, the macro time step size h must be sufficiently small in order to enable stable co-simulation. Second, for stable co-simulation, the mass ratio $\frac{m_R}{m_F^{co}}$ must be greater than a certain threshold, independent of the macro time step size h . Moreover, the threshold becomes more restrictive with increasing order of extrapolation.

The characteristic polynomial of \mathcal{S}_0 in the test problem is given as

$$0 = \mu^{4r+3} \left(\mu + \frac{1}{\mu} \left(\sum_{i=0}^r \frac{\alpha_i}{\mu^i} \right)^2 \frac{m_F^{co}}{m_R} \right), \tag{14.17}$$

where α_i are the constant coefficients of the extrapolation operator $\mathcal{T}_{n_y}^r(1)$.

With constant extrapolation ($r = 0$) this leads to $\rho(\mathcal{S}_0) = \sqrt{\frac{m_F^{co}}{m_R}}$, such that $\frac{m_R}{m_F^{co}} > 1$ must hold for zero-stable simulation. Further, for linear extrapolation $\rho(\mathcal{S}_0) < 1$ is fulfilled for $\frac{m_R}{m_F^{co}} > 6.464$ and quadratic extrapolation requires $\frac{m_R}{m_F^{co}} > 31.606$ for zero-stable simulation. This is in perfect accordance with the plots in Fig. 14.4.

Additional investigations have shown, that other system parameters (stiffness and damping coefficients) have only minor influence on the stability of the co-simulation. Higher stiffnesses and less damping require slightly smaller macro time step sizes h , but the mass ratio threshold is unchanged.

In [5], similar stability considerations are performed, but for coupling of subsystems by a coupling stiffness and damping parameter, which significantly influence the stability. In contrast, coupling by constraints was investigated in [14], where the algebraic variables have been exchanged.

14.4 Influence of the Spatial Discretization on the Stability

The previous observations have shown, that the mass ratio of the coupled subsystems is decisive for a stable co-simulation. If, for the test problem, $\frac{m_R}{m_F^{co}}$ is smaller than a certain threshold, we are faced with unstable co-simulation, independent of the remaining system parameters and the macro time step size h .

One way to circumvent this problem without changing the physical properties of the system, is to refine the discretization of the flexible structure, such that m_F^{co} gets smaller and, consequently, $\frac{m_R}{m_F^{co}}$ becomes greater than the required mass ratio threshold.

In the following, we will analyze this refinement procedure, again using the test problem from before. To keep the notation simple, we consider no dissipative terms in the flexible structure.

14.4.1 Continuous Flexible Structure

Initially, a flexible structure is formulated continuously with states $q_F(s, t)$ and $s \in [0, l]$, as sketched in Fig. 14.5. We define the length l , cross section area a , density ρ and Young's modulus E . Beyond the flexible structure, i.e. for $s \notin [0, l]$, we assume that $q_F(s, t)$ vanishes.

The kinetic energy of this continuum is given as

$$W_{kin} = \int_0^l \frac{\rho a}{2} \dot{q}_F(s, t)^2 ds \tag{14.18}$$

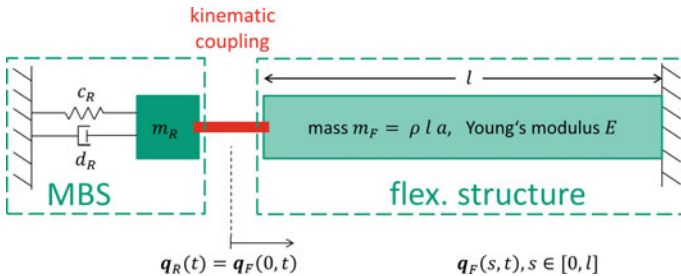


Fig. 14.5 Test model: kinematic coupling with continuous flexible structure

and for the potential energy we have

$$W_{pot} = \int_0^l \frac{Ea}{2} \mathbf{q}'_F(s, t)^2 ds, \tag{14.19}$$

where $\mathbf{q}'_F := \frac{\partial \mathbf{q}_F}{\partial s}$ denotes the spatial derivative w.r.t. s .

The corresponding energy densities are

$$\mathcal{T} = \frac{\rho a}{2} \dot{\mathbf{q}}_F(s, t)^2 \quad \text{and} \quad \mathcal{V} = \frac{Ea}{2} \mathbf{q}'_F(s, t)^2 \tag{14.20}$$

and we use the Euler–Lagrange equations

$$\frac{\partial}{\partial t} \left(\frac{\partial \mathcal{L}}{\partial \dot{\mathbf{q}}_F} \right) = \frac{\partial \mathcal{L}}{\partial \mathbf{q}_F} - \frac{\partial}{\partial s} \left(\frac{\partial \mathcal{L}}{\partial \mathbf{q}'_F} \right), \quad \mathcal{L} = \mathcal{T} - \mathcal{V}, \tag{14.21}$$

which lead to the partial differential equation

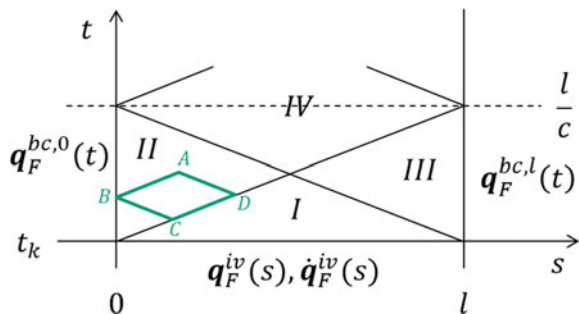
$$\rho a \ddot{\mathbf{q}}_F(s, t) - Ea \mathbf{q}''_F(s, t) = 0. \tag{14.22}$$

This is the well-known one-dimensional wave equation with propagation velocity $c = \sqrt{\frac{E}{\rho}}$. In every macro time step, we have to solve this partial differential equation with corresponding initial and boundary conditions. Figure 14.6 indicates how we can successively compute the analytical solution $\mathbf{q}_F(s, t)$, as shown e.g. in [8].

At the beginning of every macro time step $t_k \rightarrow t_{k+1}$ of the co-simulation, we update the initial values $\mathbf{q}_F^{iv}(s) = \mathbf{q}_F(s, t_k)$, $\dot{\mathbf{q}}_F^{iv}(s) = \dot{\mathbf{q}}_F(s, t_k)$, $s \in [0, l]$. Additionally, we have to construct boundary conditions $\mathbf{q}_F^{bc,0}(t)$, $\mathbf{q}_F^{bc,l}(t)$, $t \in [t_k, t_{k+1}]$, i.e. the input \mathbf{u}_F for the flexible structure, by extrapolation of the multibody system coupling state $\mathbf{q}_R(t)$.

Due to the force-displacement coupling, the coupling constraint $0 = \mathbf{q}_R(t) - \mathbf{q}_F(0, t)$ is fulfilled at previous macro time points t_k, \dots, t_{k-r} . Thus, for the interpolation polynomial $\mathbf{q}_F^{bc,0}(t)$ of order r , it holds $\mathbf{q}_F^{bc,0}(t_k) = \mathbf{q}_R(t_k) = \mathbf{q}_F(0, t_k)$. Consequently, initial values and boundary conditions are consistent, as can be seen from

Fig. 14.6 Computation scheme for successively evaluated solution $\mathbf{q}_F(s, t)$



$$\mathbf{q}_F^{iv}(0) = \mathbf{q}_F(0, t_k) \quad \text{and} \quad \mathbf{q}_F^{bc,0}(t_k) = \mathbf{q}_F(0, t_k). \quad (14.23)$$

Similarly, consistent initial values and boundary conditions are given for the boundary at $s = l$, which is fixed at the wall in this particular example.

In region *I* (cf. Fig. 14.6) the solution is determined by initial values as

$$\mathbf{q}_F(s, t_k + h) = \frac{1}{2} (\mathbf{q}_F^{iv}(s + ch) + \mathbf{q}_F^{iv}(s - ch)) + \frac{1}{2c} \int_{s-ch}^{s+ch} \dot{\mathbf{q}}_F^{iv}(\xi) d\xi. \quad (14.24)$$

Moreover, for parallelograms *ABCD* the solution of the wave equation fulfills $\mathbf{q}_F(A) = -\mathbf{q}_F(C) + \mathbf{q}_F(B) + \mathbf{q}_F(D)$. Thus, with the solution from region *I* and boundary data $\mathbf{q}_F(0, t) = \mathbf{q}_F^{bc,0}(t)$ and $\mathbf{q}_F(l, t) = \mathbf{q}_F^{bc,l}(t)$, this allows to compute the solution successively in all points in region *II*, *III* and so on.

Also, the constraint forces must be computed, or more precisely, the Lagrange multiplier $\mathbf{\Lambda}$, as input for the multibody system. Hence, we additionally regard the energy which corresponds to the coupling constraint

$$W_{con} = \int_0^l (\mathbf{q}_R(t) - \mathbf{q}_F(s, t)) \delta(s) \mathbf{\Lambda} ds, \quad (14.25)$$

where $\delta(s)$ is the Dirac delta function. If we include this energy term in the Lagrangian density $\mathcal{L} = \mathcal{T} - \mathcal{V} - (\mathbf{q}_R(t) - \mathbf{q}_F(s, t)) \delta(s) \mathbf{\Lambda}$, we arrive at the previous Euler–Lagrange equation, but with non-zero right-hand side

$$\varrho a \ddot{\mathbf{q}}_F(s, t) - Ea \mathbf{q}_F''(s, t) = \delta(s) \mathbf{\Lambda}. \quad (14.26)$$

For $\varepsilon > 0$, we can write

$$\int_{-\varepsilon}^{\varepsilon} (\varrho a \ddot{\mathbf{q}}_F(s, t) - Ea \mathbf{q}_F''(s, t)) ds = \int_{-\varepsilon}^{\varepsilon} \delta(s) \mathbf{\Lambda} ds = \mathbf{\Lambda} \quad (14.27)$$

and thus it holds (since $Ea \mathbf{q}_F'(-\varepsilon, t) = 0$ because $\mathbf{q}_F(s, t) = 0$ for $s \notin [0, l]$)

$$\mathbf{\Lambda} = \varrho a \int_{-\varepsilon}^{\varepsilon} \ddot{\mathbf{q}}_F(s, t) ds - Ea \mathbf{q}_F'(\varepsilon, t) \xrightarrow{\varepsilon \rightarrow 0} -Ea \mathbf{q}_F'(0, t). \quad (14.28)$$

In the output

$$\mathbf{y}_F = \mathbf{\Lambda} = -Ea \mathbf{q}_F'(0, t) \quad (14.29)$$

no explicit dependence on the input \mathbf{u}_F is given, i.e. no algebraic loop exists in the input-output relation of the coupled subsystems. The next section will show, that the algebraic loop appears due to the spatial discretization of the flexible structure and leads to unstable co-simulation.

14.4.2 Equidistant Discretization of the Flexible Structure

Now, as depicted in Fig. 14.7, the flexible structure is discretized equidistantly and consists of the coupling state \mathbf{q}_F^{co} and N inner states $\mathbf{q}_F^{in,1}, \dots, \mathbf{q}_F^{in,N}$. The multibody system (14.15a) and its output (14.15b) is unchanged.

For the flexible structure, the discrete energy terms for $N + 1$ states and a corresponding grid step size $\Delta s = \frac{l}{N+1}$ are

$$W_{kin} = \frac{\rho a}{2} \Delta s (\dot{\mathbf{q}}_F^{co})^2 + \sum_{n=1}^N \frac{\rho a}{2} \Delta s (\dot{\mathbf{q}}_F^{in,n})^2, \tag{14.30a}$$

$$W_{pot} = \frac{Ea}{2} \Delta s \left(\frac{\mathbf{q}_F^{in,1} - \mathbf{q}_F^{co}}{\Delta s} \right)^2 + \frac{Ea}{2} \Delta s \left(\frac{-\mathbf{q}_F^{in,N}}{\Delta s} \right)^2 + \sum_{n=1}^{N-1} \frac{Ea}{2} \Delta s \left(\frac{\mathbf{q}_F^{in,n+1} - \mathbf{q}_F^{in,n}}{\Delta s} \right)^2, \tag{14.30b}$$

$$W_{con} = (\mathbf{q}_R - \mathbf{q}_F^{co}) \Lambda. \tag{14.30c}$$

Consequently, with $\mathcal{L} = W_{kin} - W_{pot} - W_{con}$ the Euler–Lagrange equations

$$0 = \frac{\partial}{\partial t} \left(\frac{\partial \mathcal{L}}{\partial \dot{\boldsymbol{\alpha}}} \right) - \frac{\partial \mathcal{L}}{\partial \boldsymbol{\alpha}}, \quad \text{with } \boldsymbol{\alpha} \in \{\mathbf{q}_F^{co}, \mathbf{q}_F^{in,1}, \dots, \mathbf{q}_F^{in,N}\} \tag{14.31}$$

result (using $\mathbf{u}_F = [\mathbf{q}_F^{co}, \mathbf{v}_F^{co}, \dot{\mathbf{v}}_F^{co}]^T$) in the equations of motion

$$\rho a \Delta s \mathbf{u}_F^{(3)} = \frac{Ea}{\Delta s} (\mathbf{q}_F^{in,1} - \mathbf{u}_F^{(1)}) + \Lambda, \tag{14.32a}$$

$$\rho a \Delta s \dot{\mathbf{v}}_F^{in,n} = \frac{Ea}{\Delta s} (\mathbf{q}_F^{in,n+1} - \mathbf{q}_F^{in,n}) - \frac{Ea}{\Delta s} (\mathbf{q}_F^{in,n} - \mathbf{q}_F^{in,n-1}) \tag{14.32b}$$

for $n = 1, \dots, N$. (It holds $\mathbf{q}_F^{in,0} = \mathbf{q}_F^{co} = \mathbf{u}_F^{(1)}$ and $\mathbf{q}_F^{in,N+1} = 0$.)

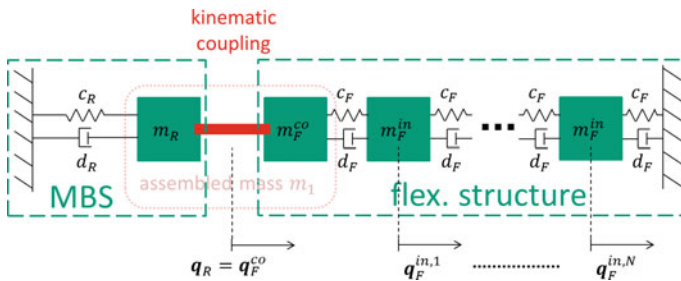


Fig. 14.7 Test model: refined $(N + 1)$ -mass spring-damper with kinematic coupling

As before, due to the coupling, Eq. (14.32a) can be skipped and is only used to compute the Lagrange multiplier

$$\mathbf{A} = \frac{\varrho a l}{N + 1} \mathbf{u}_F^{(3)} - \frac{E a (N + 1)}{l} (\mathbf{q}_F^{in,1} - \mathbf{u}_F^{(1)}). \tag{14.33}$$

Thus, the output $\mathbf{y}_F = \mathbf{A}$ of the discretized flexible structure explicitly depends on its input \mathbf{u}_F and causes an algebraic loop. For $N \rightarrow \infty$, (14.33) converges to the computation of \mathbf{A} from the continuous case

$$\mathbf{A} \xrightarrow{N \rightarrow \infty} -Ea \lim_{\Delta s \rightarrow 0} \frac{\mathbf{q}_F^{in,1} - \mathbf{u}_F^{(1)}}{\Delta s}. \tag{14.34}$$

For constant total mass, we vary the spatial discretization, i.e. N , and check $\rho(\mathcal{S}(h))$. The results are shown in Fig. 14.8. Independent of the macro time step size h , unstable simulation has to be expected for coarse spatial discretization. A finer resolution, together with sufficiently small macro time step size h , leads to stable simulation, which can also be seen from the zero-stability analysis ($h \rightarrow 0$), e.g., for constant extrapolation

$$\rho(\mathbf{DL}) = \sqrt{\frac{\varrho a \Delta s}{m_R}} = \sqrt{\frac{\varrho a l}{(N + 1)m_R}} \xrightarrow{N \rightarrow \infty} 0. \tag{14.35}$$

At the same time, the stiffness coefficients

$$c_F = \frac{E a}{\Delta s} = \frac{E a (N + 1)}{l} \tag{14.36}$$

increase for $N \rightarrow \infty$ and slightly smaller macro time steps are required for stable simulations. This can also be observed from the stability regions in Fig. 14.8.

To confirm the stability thresholds, we perform some numerical experiments. If linear extrapolation is used, we deduce a critical number of discrete flexible inner states $N \approx 25$ from Fig. 14.8. Further, the macro time step size $h = 10^{-2}$ s should be sufficient and is used in the following.

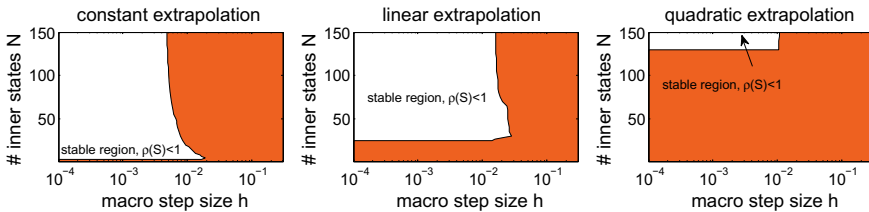


Fig. 14.8 Stability regions depending on the spatial discretization (number of inner states N) and the macro time step size h with constant, linear and quadratic extrapolation

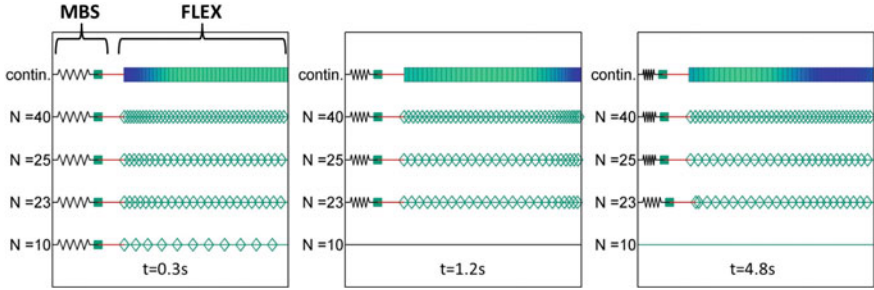


Fig. 14.9 Snapshots at discrete points in time of co-simulation with varying spatial discretization (number of inner states N) of the flexible structure

Fig. 14.10 Motion of the rigid body state q_R for varying spatial discretization (number of inner states N) of the flexible structure

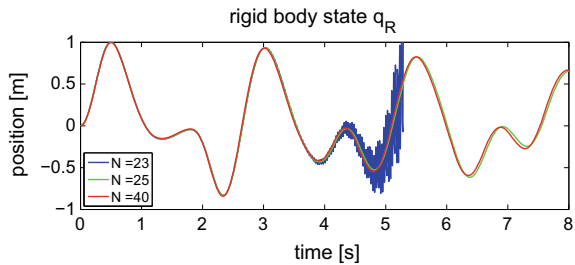


Figure 14.9 visualizes the simulations with varying spatial discretization. They are performed for $N = 10, 23, 25, 40$ and the continuous case. At time $t = 1.2$ s, the simulation with $N = 10$ already failed. The simulation with $N = 23$ shows small deviations at time $t = 4.8$ s, while for $N = 25$ and $N = 40$ no unstable behavior is observable, as expected from the stability plots.

Additionally, Fig. 14.10 shows the motion of the rigid body state q_R . As long as stability is given (for $N = 25$ and $N = 40$), the motion is almost equal for different discretizations. For $N = 23$, the simulation starts to oscillate approximately at time $t = 4.0$ s.

14.4.3 Non-equidistant Discretization of the Flexible Structure

If a very fine discretization is necessary to achieve stable co-simulation, this also leads to a strongly increasing number of degrees of freedom and may result in long computation time. However, it is sufficient to refine the discretization of the flexible structure only at the coupling interface, i.e. we use a non-equidistant spatial discretization. Especially, only one additional inner state can be used as depicted in Fig. 14.11. A small part of the original coupling mass is used for the coupling, while the remaining part becomes the corresponding mass of the additional inner state.

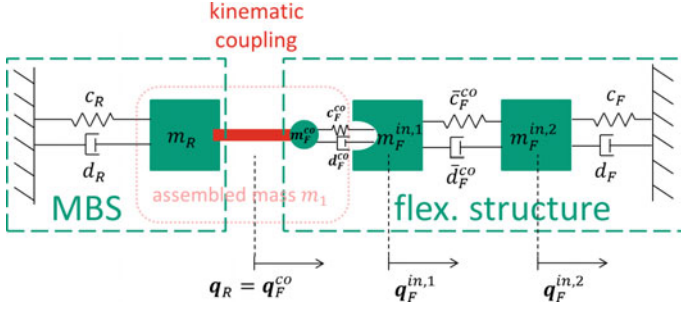


Fig. 14.11 Test model: 2-mass spring-damper with fractional coupling state

For simplicity, we start from the original 2-mass spring-damper model (i.e. it holds $N = 1$, but also arbitrary N is possible), where Δs is the original discrete grid step size in space from the equidistant case.

At the grid next to the coupling interface, we take a fraction $\theta \in (0, \frac{1}{2}]$ from Δs to build a refined discretization with step sizes $\theta\Delta s$ and $(1 - \theta)\Delta s$. Moreover, if we concentrate the mass in the grid points, this leads to

$$m_F^{co} = \rho a \frac{\theta \Delta s}{2}, \quad m_F^{in,1} = \rho a \frac{\Delta s}{2} \quad \text{and} \quad m_F^{in,2} = \rho a \frac{(2 - \theta) \Delta s}{2}. \quad (14.37)$$

With input $\mathbf{u}_F = [\mathbf{q}_F^{co}, \mathbf{v}_F^{co}, \dot{\mathbf{v}}_F^{co}]^T$ the equations of motion are

$$m_F^{co} \mathbf{u}_F^{(3)} = \frac{Ea}{\theta \Delta s} (\mathbf{q}_F^{in,1} - \mathbf{u}_F^{(1)}) + \mathbf{A}, \quad (14.38a)$$

$$m_F^{in,1} \dot{\mathbf{v}}_F^{in,1} = -\frac{Ea}{\theta \Delta s} (\mathbf{q}_F^{in,1} - \mathbf{u}_F^{(1)}) + \frac{Ea}{(1 - \theta) \Delta s} (\mathbf{q}_F^{in,2} - \mathbf{q}_F^{in,1}), \quad (14.38b)$$

$$m_F^{in,2} \dot{\mathbf{v}}_F^{in,2} = -\frac{Ea}{(1 - \theta) \Delta s} (\mathbf{q}_F^{in,2} - \mathbf{q}_F^{in,1}) + \frac{Ea}{\Delta s} (0 - \mathbf{q}_F^{in,2}). \quad (14.38c)$$

From (14.38a), the output $\mathbf{y}_F = \mathbf{A}$ is computed

$$\mathbf{A} = \frac{\theta \rho a \Delta s}{2} \mathbf{u}_F^{(3)} - \frac{Ea}{\theta \Delta s} (\mathbf{q}_F^{in,1} - \mathbf{u}_F^{(1)}) \quad (14.39)$$

and explicitly depends on the input \mathbf{u}_F , again causing an algebraic loop. For $\theta \rightarrow 0$, this also converges to the computation of \mathbf{A} from the continuous case

$$\mathbf{A} \xrightarrow{\theta \rightarrow 0} -Ea \lim_{\theta \rightarrow 0} \frac{\mathbf{q}_F^{in,1} - \mathbf{u}_F^{(1)}}{\theta \Delta s}. \quad (14.40)$$

Similar to the case of equidistant discretization and $N \rightarrow \infty$, the non-equidistant discretization behaves for $\theta \rightarrow 0$, since the coupling mass $m_F^{co} = \rho a \frac{\theta \Delta s}{2}$ tends to zero and the leftmost stiffness $\frac{Ea}{\Delta s \theta}$ increases for $\theta \rightarrow 0$.

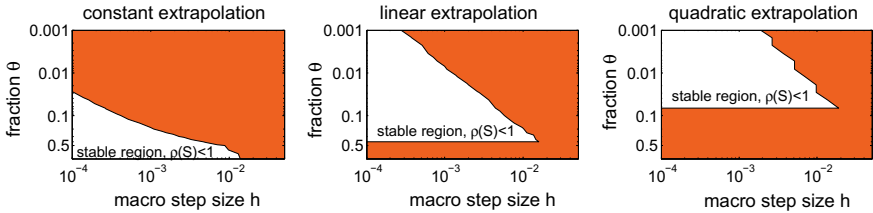


Fig. 14.12 Stability regions depending on the spatial discretization ($N = 10$ and varying fraction θ) and the macro time step size h with constant, linear and quadratic extrapolation

Fig. 14.13 Motion of the rigid body state q_R for varying spatial discretization ($N = 10$ and varying fraction θ) of the flexible structure

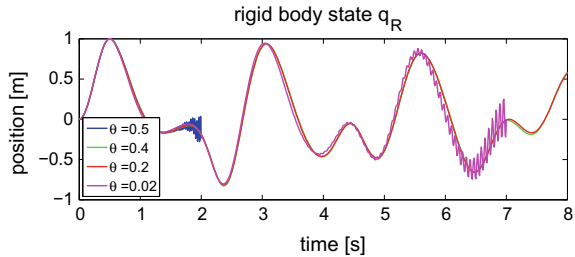
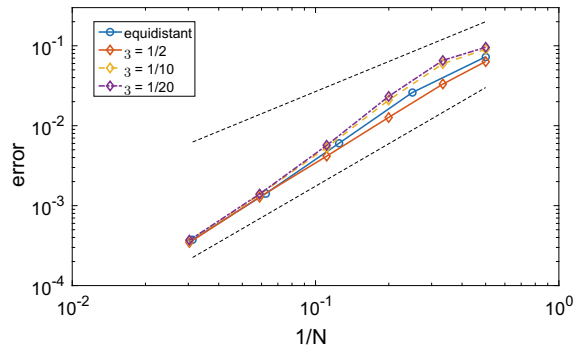


Fig. 14.14 Error of the flexible structure for equidistant and non-equidistant discretization depending on N



Stability regions for $N = 10$ and varying θ are shown in Fig. 14.12. If the fraction θ is sufficiently small, the simulations are zero-stable. At the same time, small fractions θ require small macro time step sizes. We want to remark, that due to the logarithmic vertical scale in Fig. 14.12, the results seem worse than for the equidistant case. However, the fraction θ does not need to be unnecessarily small, such that quite large macro time step sizes h can still be chosen.

Also here, numerical experiments with linear extrapolation and $h = 10^{-2}$ s confirm the theoretical stability analysis (see Fig. 14.13). With $\theta = 0.5$ the co-simulation is still unstable, while with $\theta = 0.4$ we get stable results. A very small fraction $\theta = 0.02$ would require a smaller macro time step size h .

Nevertheless, compared with the equidistant case, we lose accuracy. In particular, only accuracy of order one is achieved locally (close to the coupling interface) instead of order two. This can be seen from a straightforward Taylor expansion.

Figure 14.14 confirms the theoretical considerations. As expected, the error for equidistant discretization is approximately of order two. The non-equidistant discretization with $\theta = \frac{1}{2}$ shows almost the same behavior. For smaller fractions θ , leading to large deviations in the grid step size, and coarse discretization, i.e. small N , a lower slope can be observed. However, accuracy of order two is achieved asymptotically for $N \rightarrow \infty$. Since only locally (at two grid points close to the coupling interface) the discretization error is larger, its influence vanishes for fine discretization.

14.5 Application Problem

We transfer the stability analysis to a more realistic application example. In order to do that, we couple a small nonlinear multibody system and a cable model. The latter is a geometrically exact Cosserat rod as described in [10].

The multibody system is simulated using local joint coordinates η_1 and η_2 , as depicted in Fig. 14.15. System parameters are given in Table 14.2.

At the lower mass m_2 , one end of the cable is coupled and all degrees of freedom are constrained. The other end is fixed at the wall. A staggered grid is used to discretize the Cosserat rod (see Fig. 14.16). The cable material is modeled as rubber, with corresponding properties: density $\rho = 1100 \frac{\text{kg}}{\text{m}^3}$, Young's modulus $E = 5 \cdot 10^6 \frac{\text{N}}{\text{m}^2}$,

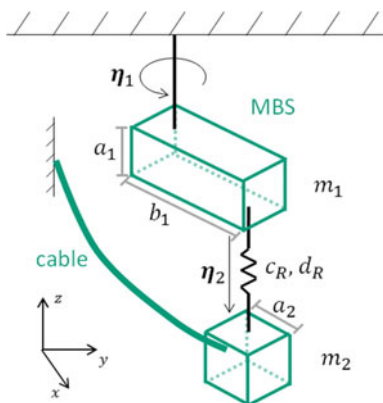


Fig. 14.15 Sketch of small multibody system and coupled cable model

Table 14.2 System parameters for the small multibody system

Param.	m_1	m_2	c_R	d_R	a_1	b_1	a_2
Value	0.5 kg	0.1 kg	5 N/m	0.1 Ns/m	0.2 m	0.4 m	0.2 m

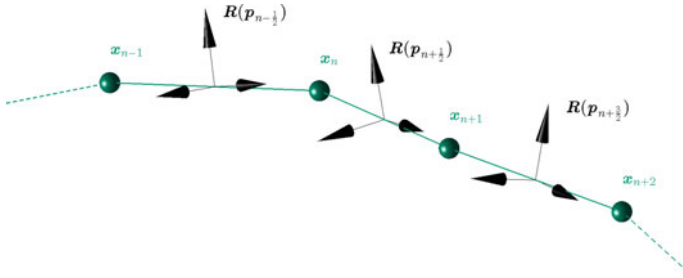


Fig. 14.16 Discrete Cosserat rod, discretized with a staggered grid: translatory degrees of freedom in the nodes, rotatory degrees of freedom on the edges

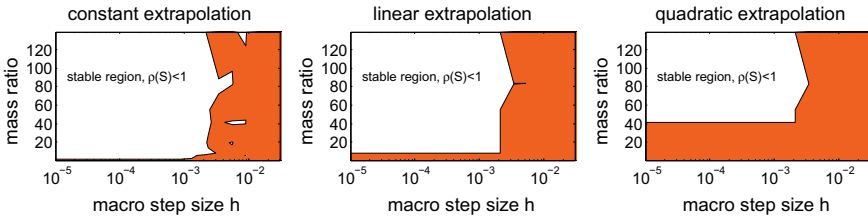


Fig. 14.17 Stability regions depending on the mass ratio “ $\frac{m_R}{m_F}$ ” and the macro time step size h with constant, linear and quadratic extrapolation

Poisson’s ratio $\nu = 0.5$, length $L = 1$ m and radius $r = 0.005$ m. The density is varied in the following analysis to modify the mass ratio “ $\frac{m_R}{m_F}$ ”.

After linearizing the subsystem equations in an equilibrium point of the coupled system, the spectral radius of $\mathcal{S}(h)$ can be computed.

Figure 14.17 shows the corresponding stability regions depending on the mass ratio and the macro time step size h for constant, linear and quadratic extrapolation. In all cases, a macro time step size $h = 10^{-3}$ s seems sufficient. However, for stable co-simulation also the mass ratio must be greater than a certain threshold. An increasing order of extrapolation requires a larger mass ratio (the mass ratio for constant extrapolation is small and hard to see here).

The observations are similar to the ones from the stability analysis of the test problem. Nevertheless, one should point out that since we consider the linearized systems here, the thresholds probably are less reliable than before.

Next, we vary the number of elements, i.e. we modify the spatial discretization of the cable. For the above analysis, the flexible structure was discretized with 12 elements.

In the following we set the density to $\varrho = 2200$ kg/m³ and consider $\mathcal{S}(h)$ for cables with 4, 6, 8, 10 and 12 elements. The stability plots are shown in Fig. 14.18.

With constant extrapolation, even 4 elements are sufficient for zero-stable co-simulation, while more than 12 elements are required using quadratic extrapolation. Thus, no thresholds for the number of elements are visible in the plots for constant

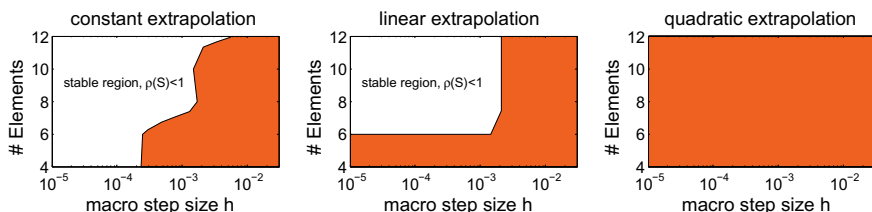


Fig. 14.18 Stability regions depending on the number of elements and the macro time step size h with constant, linear and quadratic extrapolation

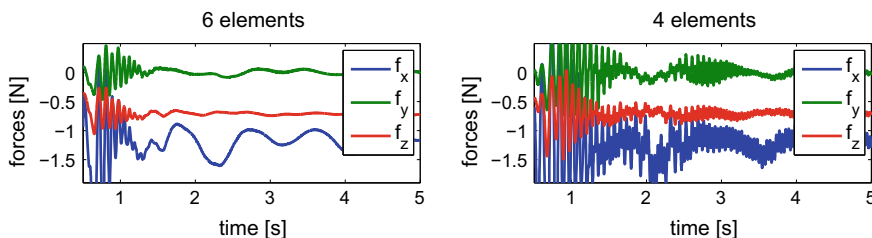


Fig. 14.19 Coupling forces for co-simulation with $h = 10^{-3}$ s and linear extrapolation. The cable is discretized with 6 elements and 4 elements

and quadratic extrapolation. Only the plot for linear extrapolation shows a threshold for the required number of elements. In particular, a coarse discretization with only 4 elements should lead to unstable co-simulation, but for 6 or more elements in combination with sufficiently small macro time step sizes h we expect stable co-simulation.

To check these conclusions, numerical experiments are performed. Figure 14.19 shows the coupling forces, i.e. the output of the flexible structure. From that, we deduce that with 6 elements the coupling forces show some oscillations but then quickly stabilize, while with 4 elements strong oscillations are present until the end of the simulation.

Hence, also for complex nonlinear subsystems, the results of the stability analysis are in good accordance with numerical experiments and can be used to design a stable co-simulation.

14.6 Conclusion

We investigated a general approach for the kinematic coupling of mechanical subsystems. In particular, a force-displacement co-simulation is derived from a constraint coupling. This allows to formulate very general coupling joints without problem-specific simplifications. The assumption that $\frac{\partial g_{co}}{\partial q_{co}^T}$ is quadratic and non-singular holds

for most practically relevant couplings (e.g. full clamping or fixed point but free rotation).

For this co-simulation approach, the stability was analyzed and a certain mass ratio of the coupled subsystem masses turns out to be decisive. If this mass ratio violates a required threshold, an algebraic loop in the formulation causes unstable behavior, independent of the chosen macro time step size.

However, this mass ratio can be adjusted by modifying the spatial discretization of the flexible structure. A finer resolution improves the mass ratio and enables stable co-simulation. In fact, it becomes clear that the algebraic loop is introduced by the spatial discretization of the flexible structure.

Finally, we considered locally refined spatial discretization. Indeed, only one additional degree of freedom close to the coupling interface is sufficient. Thus, stable co-simulation can be achieved without much additional effort.

References

1. Arnold, M., Günther, M.: Preconditioned dynamic iteration for coupled differential-algebraic systems. *BIT Numer. Math.* **41**(1), 001–025 (2001)
2. Bartel, A., Brunk, M., Günther, M., Schöps, S.: Dynamic iteration for coupled problems of electric circuits and distributed devices. *SIAM J. Sci. Comput.* **35**(2), B315–B335 (2013)
3. Bazilevs, Y., Hsu, M.C., Zhang, Y., Wang, W., Liang, X., Kvamsdal, T., Brekken, R., Isaksen, J.: A fully-coupled fluid-structure interaction simulation of cerebral aneurysms. *Comput. Mech.* **46**(1), 3–16 (2010)
4. Busch, M.: Zur effizienten kopplung von simulationsprogrammen. Ph.D. thesis, Universität Kassel (2012)
5. Busch, M., Schweizer, B.: Numerical stability and accuracy of different co-simulation techniques: analytical investigations based on a 2-dof test model. In: *The 1st Joint International Conference on Multibody System Dynamics*, Lappeenranta, Finland, 25–27 May 2010
6. Felippa, C.A., Park, K., Farhat, C.: Partitioned analysis of coupled mechanical systems. *Comput. Methods Appl. Mech. Eng.* **190**(24), 3247–3270 (2001)
7. FMI: Functional Mock-up Interface for Model Exchange and Co-Simulation Version 2.0. <https://www.fmi-standard.org/downloads>
8. John, F.: *Partial Differential Equations*, Applied Mathematical Sciences, vol. 1. Springer, New York (1982)
9. Kübler, R., Schiehlen, W.: Two methods of simulator coupling. *Math. Comput. Model. Dyn. Syst.* **6**(2), 93–113 (2000)
10. Lang, H., Linn, J., Arnold, M.: Multibody dynamics simulation of geometrically exact Cosserat rods. *Multibody Syst. Dyn.* **25**(3), 285–312 (2011)
11. Roberson, R.E., Schwertassek, R.: *Dynamics of Multibody Systems*. Springer, Berlin (1988)
12. Schneider, F.: A differential-algebraic coupling approach for force-displacement co-simulation of flexible multibody systems with kinematic coupling. Ph.D. thesis, Technische Universität Kaiserslautern (2016)
13. Schneider, F., Burger, M., Arnold, M., Simeon, B.: A new approach for force-displacement co-simulation using kinematic coupling constraints. *Z. Angew. Math. Mech.* **97**(9), 1147–1166 (2017)
14. Schweizer, B., Lu, D.: Stabilized index-2 co-simulation approach for solver coupling with algebraic constraints. *Multibody Syst. Dyn.* **34**(2), 129–161 (2015)
15. Simeon, B.: *Computational Flexible Multibody Dynamics*. Springer, Berlin (2013)

16. Wehage, R., Haug, E.: Generalized coordinate partitioning for dimension reduction in analysis of constrained dynamic systems. *J. Mech. Des.* **104**(1), 247–255 (1982)
17. Woernle, C.: *Mehrkörpersysteme*. Springer, Berlin (2011)

Author Index

A

Arnold, Martin, [99](#)

B

Benedikt, Martin, [1](#)

Braun, Robert, [27](#)

Burger, Michael, [43](#), [267](#)

Busch, Martin, [57](#)

D

Drenth, Edo, [1](#)

F

Fritzson, Dag, [27](#)

G

Gomes, Cláudio, [81](#)

H

Hällqvist, Robert, [27](#)

Hante, Stefan, [99](#)

Horie, Tomoyoshi, [117](#)

I

Ishihara, Daisuke, [117](#)

J

Jungers, Raphaël M., [81](#)

K

Köbis, Markus, [99](#)

Kouroussis, Georges, [241](#)

Kraft, Jan, [131](#), [217](#)

L

Legat, Benoît, [81](#)

Li, Pu, [153](#)

Lu, Daixing, [153](#), [217](#)

M

Meijaard, J. P., [203](#)

Meyer, Tobias, [131](#), [217](#)

Morawietz, Sissy, [253](#)

N

Niho, Tomoya, [117](#)

O

Olivier, Bryan, [241](#)

S

Schauer, Marco, [253](#)

Schmoll, Robert, [153](#)

Schneider, Fabio, [267](#)

Schweizer, Bernhard, [131](#), [153](#), [217](#)

Steidel, Stefan, [43](#)

T

Taddei, Francesca, [253](#)

V

Vangheluwe, Hans, [81](#)

Verlinden, Olivier, [241](#)